Escuela Politécnica Superior

Departamento de Ingeniería Informática

A semantic-based framework for building cross-domain networks: Application to item recommendation

Ignacio Fernández Tobías

under the supervision of

Iván Cantador Gutiérrez

Master thesis

Programa Oficial de Posgrado en Ingeniería Informática y de Telecomunicación.

Escuela Politécnica Superior, Universidad Autónoma de Madrid

September 2013

Abstract

Recommender systems have been successfully used in a large number of applications on the Web, to cope with the information overload problem by finding the items –e.g. movies, music compositions, books, and news articles– that best suit the users' preferences –tastes and interests– without requiring explicit queries to be launched, as usually done in information retrieval systems.

Most of the developed recommender systems target items that belong to a single domain. For instance, Netflix suggests a user with movies and TV series, and Last.fm makes personalized recommendations of music artists and compositions. Nowadays, nonetheless, there are many leisure and e-commerce Web sites, like Amazon, which could take benefit from exploiting user preferences on diverse types of items to provide recommendations in different but somehow related domains. Recommendations across domains could mitigate the cold-start problem when little information about the user's preferences is available in a target domain, and are potentially more diverse and serendipitous than single-domain recommendations.

The goal of cross-domain recommender systems is to suggest items in a target domain by exploiting user preferences and/or domain knowledge available in a distinct source domain. As a fairly recent research topic, so far cross-domain recommendation has been mostly addressed in the collaborative filtering setting, where there is some user preference (rating) overlap between domains. But this may be unlikely in realistic situations, and hence, a major challenge for making cross-domain recommendations is to establish methods for transferring knowledge across domains, in which usually there is little or no overlap between user preferences or item attributes.

In this thesis, we present a framework for automatically building cross-domain semantic networks using structured information extracted from Linked Data repositories. These networks connect concepts from several domains through semantic properties, thus establishing bridges that can be used to support or perform cross-domain recommendations. Upon the built semantic networks, we investigate graph-based methods to compute the semantic relatedness between items from different domains, and study rank aggregation techniques and personalized graph ranking algorithms to generate semantically enhanced cross-domain recommendations.

We instantiated our framework for the case study of suggesting music artists and compositions semantically related to places of interest, using DBpedia –i.e., the Wikipedia's ontology– as multi-domain knowledge base. For this particular application, we validated our approach through two sets of experiments. First, we computed the accuracy of the proposed semantic relatedness metrics, and evaluated the capability of our framework to find musicians related to places of interest. Second, we analyzed whether users prefer semantically enhanced recommendations over purely content-based recommendations. We found that using a constrained spreading algorithm we were able to achieve between 70% and 90% accuracy in finding music artists and compositions semantically related to places of interest, and that users tend to choose as relevant musicians that are related to places, instead of musicians that fit the users' music preferences.

Acknowledgements

The work presented here is the result of two years of Master studies under the supervision of Dr. Iván Cantador. I am very grateful to him not only for guiding and encouraging me in the confusing stage of starting a research career, but also for all his interest and hard work, and for his continuous help and patience. I am also very grateful to Dr. Marius Kaminskas and Prof. Francesco Ricci, who collaborated closely in this work, for all their contributions and valuable feedback. I would also like to thank all the people from the Information Retrieval Group at Universidad Autónoma de Madrid for all the things I have learnt from them, and for the always interesting discussions we had.

Finally, I would like to specially thank my family for their support and for putting up with me this whole time. I am sure sometimes it was not easy. This work is dedicated to them.

Contents

Chapter 1. Introduction

1.1 Motivation

Recommender systems (RSs) are a type of information filtering systems that aim to cope with the information overload problem by finding the items –e.g. movies, music compositions, books, and news articles– that best suit the users' preferences –tastes and interests– (Adomavicius & Tuzhilin, 2005), without requiring explicit queries to be launched, as usually done in information retrieval systems. Users thus can easily filter items that are of low relevance or utility, and focus only on a small selection of (potentially) relevant items.

In the last years, RSs have been successfully used in a large number of e-commerce and entertainment Web sites, such as Amazon[1], Netflix[2], YouTube[3], Last.fm[4], BookCrossing[5] and Google News[6], to name a few. The vast majority of these systems, however, only offer recommendations of items belonging to a single domain. Hence, for example, Netflix suggests a user with movies and TV series, and Last.fm makes personalized recommendations of music artists and compositions. In both cases, the recommendations are computed using user feedback (ratings) about items in the target domain. This is not perceived as a limitation in such sites, since they focus on a precise domain/market. But in others, like Amazon e-commerce site, it would be useful and meaningful to make recommendations on diverse types of items, and to elaborate recommendations of items of a particular type –e.g. recommendations of music compositions– on the base of the full set of the user's preferences on other somehow related items of different types –such as movies and books–, or because the "context" of the user is expressed in a different but somehow related domain –such as the place the user is visiting. In fact, there are dependencies and correlations between user preferences in different domains, as market studies and classical data mining researches have revealed (Stupar & Michel, 2011; Winoto & Tang, 2008). Therefore, for recommendation purposes, instead of treating each domain independently, knowledge acquired in one domain could be transferred and exploited in other domains.

Some RSs already offer recommendations of items from different domains, but in general, in order to build a recommendation of an item in a particular target domain, they exploit only user information –i.e., preferences, context– on that domain or, considering a cross-domain situation, they apply collaborative filtering (CF) techniques (Desrosiers & Karypis, 2011; Herlocker et al., 1999) on datasets in which there is some user preference (rating) overlap between domains. In both cases, the systems do not need information about item attributes, and avoid the difficulty of dealing with existing data heterogeneity.

Among the existing CF techniques, traditional neighborhood-based models applied on the single domain recommendation problem have been adapted to the multiple domain case (Berkovsky et al., 2007; Tiroshi & Kuflik, 2012; Winoto & Tang, 2008). In this case, aiming to select the neighbors accurately, it is necessary to have an enough user preference overlap between the involved domains, and a clear indication about whether users with similar preferences in one domain also have likewise preferences in other domain.

---

1 Amazon e-commerce site, http://www.amazon.com
2 Netflix on-demand streaming media site, http://www.netflix.com
3 YouTube video sharing site, https://www.youtube.com
4 Last.fm on-line radio, http://www.last.fm
5 BookCrossing on-line book exchange site, http://bookcrossing.com
6 Google News on-line news aggregator, http://news.google.com

Despite these limitations, little work has been done outside the CF framework, attempting e.g. to make use of the items' attributes. Nonetheless, we believe that knowledge-based recommender systems, which exploit explicit knowledge about users and items (Burke, 2000), are a compelling approach for reasoning across domains, and providing item suggestions on this basis. But, as stated by Felfernig et al. (2011), knowledge-based recommendation approaches suffer from the so-called knowledge acquisition bottleneck, in the sense that the exploited knowledge has to be converted into formal, executable representations; and thus, even if cross-domain knowledge can be appropriate for solving recommendation tasks, it has to be properly collected, processed and modeled.

With the advent of the Semantic Web, and specifically with its reference implementation in the Linked Data initiative (Bizer et al., 2009), new opportunities arise to face the above difficulties. A very large and increasing number of structured knowledge sources and ontologies extend the Web with a global data space connecting data from diverse domains such as people, organizations, music, movies, books and reviews; and new types of recommendation strategies capable of handling item heterogeneity can now be developed by exploiting the Linked Data repositories together with their explicit semantic relations within and between them.

In this master thesis, we state the **hypothesis** that cross-domain semantic networks built by mining Linked Data can enhance recommendations in a target domain by incorporating information about the user's preferences and contexts from different, related source domains. To validate such hypothesis, we develop and evaluate a framework for building such semantic networks, and exploiting them to generate cross-domain item recommendations.

1.2 Objective and research questions

The main **objective** of this research is to develop a framework upon which items from different domains are connected by means of semantic relations, shaping a semantic network that can be used to generate cross-domain item recommendations taking user preferences and contextual signals into consideration.

To achieve this objective, we investigate a graph-based knowledge representation, and a data extraction method for Linked Data repositories, which let (semi-)automatically build the above cross-domain semantic network. Based on the built network, we also study semantic relatedness metrics to quantify the strength of inter-domain relations, and several graph-based ranking algorithms, which let provide recommendations for items in a target domain exploiting user preferences and contextual signals associated to items in a source domain.

The proposed framework relies on the following **research questions**, addressed in this thesis:

- **RQ1. How to uniformly represent semantic relations between items from different arbitrary domains?**

  To handle the heterogeneity of items belonging to distinct domains, we shall define a generic knowledge representation model based on semantic graphs/networks.

- **RQ2. How to automatically identify items in a target domain that are related to items in a source domain?**

  To establish relations between items belonging to different domains, we shall exploit multi-domain information sources, such as the Linked Data repositories, and shall address the so-called semantic data acquisition bottleneck problem, i.e., collecting and transforming knowledge into a formal and executable representation.

- **RQ3. How to measure and evaluate the semantic relatedness between two items belonging to different domains?**

  To identify the items in a target domain that best fit certain items in a source domain, we shall develop metrics to measure how meaningful the established semantic relations between items are, and shall evaluate them by means of user studies.

- **RQ4. Can semantically related items be used to enhance cross-domain recommendations based on user preferences?**

  To prove if users actually find recommendations of semantically related items relevant and useful, or if, in contrast, they just prefer recommendations that only rely on user preferences, we shall conduct a user study evaluating both types of recommendations.

To address these questions, for the sake of simplicity, and without losing generality, in this thesis we shall implement and evaluate the proposed framework for a particular context-aware recommendation problem –**suggesting music artists and composition for places of interest**–, where we have to provide personalized recommendations of items in a target domain (music) when the user's context is modeled with items in a different, related source domain (places of interest).

In the next section we briefly introduce the proposed framework, which consists of four stages aimed to address the stated research questions.

1.3 Overview of the proposed approach

As stated previously, the goal of this thesis is to develop a framework for automatically building semantic networks that connect items from different domains through explicit relations, and exploit such networks to provide recommendations of items in a target domain considering the user's preferences and/or context in a distinct source domain. Our approach consists of 4 stages, which address each of the research questions stated in Section 1.2, and are depicted in Figure 1.

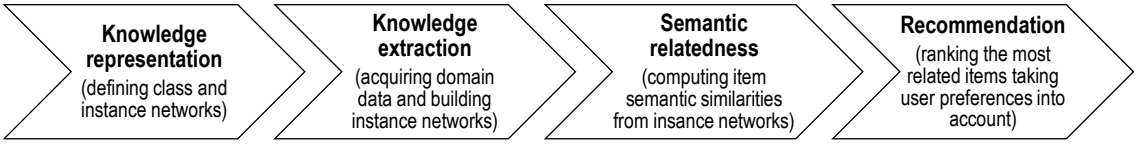

Figure 1. Stages of the proposed approach.

In the following we briefly describe each of the stages. We provide full descriptions of them in Chapter 3.

Stage 1. Knowledge representation

In our approach, we represent the knowledge describing and relating source and target domains by means of semantic networks linking concepts –which correspond to either **classes** (e.g. "*Book*") or **instances** (e.g. "*Rita Hayworth and The Shawshank Redemption*", as an instance of the class Book)– through **properties** (e.g. Rita Hayworth and The Shawshank Redemption "*was written by*" Stephen King, being Stephen King an instance of a class Writer).

The classes and properties considered by our approach for particular domains are identified in Linked Data repositories. For example, *movies* and *books* could be linked directly because a given movie *is based on* certain book, or indirectly because the book *was written by* an author who also *wrote the script for* the movie. Once identified, the classes and relations are naturally arranged into a semantic graph/network. In this **class-level network**, there are root nodes with no incoming edges

that are associated to items belonging to the source domain, and sink nodes without outgoing edges that represent the items in the target domain.

Once the class-level network has been defined, we instantiate it for a particular item belonging to a root class (in the source domain), by extracting domain knowledge from Linked Data repositories. Thus, an **instance-level network** is automatically generated. In the example given before, the corresponding instance network may connect a movie node "*The Shawshank Redemption*" with a book node "*Rita Hayworth and The Shawshank Redemption*" through an edge associated to the property "*is based on*."

Stage 2. Knowledge extraction

Our framework relies on the connected Linked Data repositories as data sources of extensive, multi-domain knowledge, in order to find relations between heterogeneous items. Since the link structure of these data sources is complex, we need a method to extract from them limited ontologies that preserve the inter-domain connections of interest. From the set of top-level classes and relevant properties previously defined in the class-level network, we perform an **offline** Depth-First search on a restricted Linked Data sub-graph to discover *domain instances* that connect the domains of interest, and which will be added to the instance-level networks.

Afterwards, aiming to find *semantically related instances* of a particular input instance in the source domain, and generate its instance-level network, we **online** instantiate the defined class-level network by automatically extracting (from the considered Linked Data repositories) instances connected according to the specified classes and relations. This is done by searching for matching tuples [subject, property, object] trough SPARQL[7] queries submitted to the repositories.

Stage 3. Semantic relatedness

Once an instance network is built, semantically relating items in the target domain with an input item in the source domain, we rank the former in order to estimate which of them are most relevant for the latter. Specifically, we compute similarity values $\mathrm{rel}(s,t)$ between an input item s and all the found related target items t by means of a number of **semantic relatedness metrics**. To define these metrics, we investigate both statistical methods that exploit the (description) content of the items, and topological algorithms that use the link structure of an instance network. For each input item s, the output of this stage is a ranking $(t_1: w_1, \dots, t_N: w_N)$ with the related target items t_j and their semantic relatedness values w_j, which will be used in the recommendation stage.

Stage 4. Recommendation

We finally adjust the **ranking** of semantically related items t_j taking into account the user's preferences. Specifically, we represent the user preferences as weighed components in a profile vector $\vec{u}$, and consider a content-based preference prediction $CB(\vec{u}, t_j)$. Thus, for each of the input instances s in the source domain, we combine the relatedness values w_j computed in Stage 3 with the results of the content-based prediction $CB(\vec{u}, t_j)$ to generate a final recommendation score:

$$score(s, t_j, \vec{u}) = g\left(w_j, CB(\vec{u}, t_j)\right)$$

where the utility gain function g will be defined based on the combination strategy performed, e.g. as a linear combination of the w_j values. Alternatively, the estimated preference score may be computed by means of a personalized semantic relatedness, by incorporating user preferences into the instance-level network, and adapting the algorithms from Stage 3, that is:

$$score(s, t_j, \vec{u}) = \mathrm{rel}(s, t_j, \vec{u})$$

[7] SPARQL Protocol and RDF Query Language, http://www.w3.org/TR/rdf-sparql-query

1.4 Case study

In order to validate our framework, we have implemented and evaluated it for a particular context-aware recommendation problem: **providing personalized music artist and composition suggestions for a particular place of interest** (POI), which the user is visiting or browsing information about. In this situation, the recommendation goal could be, for instance, suggesting musicians who were born in the same city where the POI is located, or musicians who lived in the same artistic period the POI belongs to. This is an interesting cross-domain application that can be exploited by useful and engaging information services. As an example, a tourism website or a mobile city guide may point to important musicians that are related to a visited place, or may play music composed by these musicians, thus providing a soundtrack to a sightseeing tour, enhancing the user's experience.

Previous work (Kaminskas & Ricci, 2011) showed that there exist latent similarities between items in the two domains –POI and music. Therefore, a match between these two types of items can be recognized, and users can perceive and appreciate the established links. In the quoted paper the matching between POIs and music compositions was computed by means of social tag-based item profiles and similarity metrics. From that result, the framework presented here enables compute matching between POIs and musicians based on their semantic relatedness. We hypothesize that given certain musicians related to a particular POI, recommendations of compositions from such musicians will fit with the place.

Particularizing the generic research questions stated in Section 1.2 to our case study, we aim to address the following research questions:

- RQ1'. What is a suitable model for representing relevant concepts in the POI and music domains, and the inter-domain semantic properties that link them?
- RQ2'. How to automatically identify musicians semantically related to a given POI?
- RQ3'. How to measure the semantic relatedness between POIs and musicians?
- RQ4'. Can musicians semantically related to a given POI be used to enhance POI-to-musician recommendations based on user music preferences?

In general, it is clearly challenging to match a POI with a music item so that users appreciate such relation, and prefer it to other music items not explicitly matching the place. The main challenge that one must face when addressing this goal is related to the fact that POIs and music items represent two different domains, and there is no obvious way to match them. Consider for instance the State Opera of Vienna, Austria. It is one of the most famous opera houses dating back to the 19th century, and a prominent attraction for tourists visiting Vienna. Music related to this place is formed by classical music pieces of composers who lived and worked in Vienna when the State Opera was erected, or who are related to the venue itself, e.g. they executed their compositions there. Performing this task automatically, for any given POI (not necessarily being a music venue), requires a way to identify the meaningful relations between POIs and music items. We propose to search for these relations in Linked Data repositories, and more specifically in DBpedia (Auer et al., 2007), which is a core ontology that connects multiple repositories from multiple domains, and is the structured version of Wikipedia[8] on-line encyclopedia.

[8] Wikipedia on-line encyclopedia, http://www.wikipedia.org

1.5 Contributions

Although we have focused on a specific case study –adapting music item recommendations to POIs– we believe that the addressed research questions have a general relevance, and the developed solutions can be applied to other domains. In fact, our framework is independent from the application domains and the used knowledge sources. Hence, we make the following general **contributions**:

- A generic method that isolates and extracts (from DBpedia) semantic networks linking items belonging to two given domains, and filters out related items of other domains.
- A framework for cross-domain item recommendation that captures semantic relatedness between items, and is flexible to incorporate and exploit user preferences and contextual signals.

1.6 Publications

Part of the work presented in this thesis has been published in the following research papers:

- Ignacio Fernández-Tobías. 2013. **Mining the Social Semantic Web for Making Cross-domain Recommendations**. In: *Proceedings of the 5th Symposium on Future Directions in Information Access (FDIA 2013)*. To appear.
- Ignacio Fernández-Tobías. 2013. **Mining Semantic Data, User Generated Contents, and Contextual Information for Cross-domain Recommendation**. In: *Proceedings of the 21st International Conference on User Modeling, Adaptation, and Personalization (UMAP 2013)*, pp. 371-375. Springer-Verlag. ISBN 978-3-642-38843-9. Doctoral consortium paper.
- Marius Kaminskas, Ignacio Fernández-Tobías, Francesco Ricci, Iván Cantador. **Ontology-based Identification of Music for Places**. In: *Proceedings of the 13th International Conference on Information and Communication Technologies in Tourism (ENTER 2013)*, pp. 436-447. Springer-Verlag. ISBN 978-3-642-36308-5. **2nd best paper award**.
- Marius Kaminskas, Ignacio Fernández-Tobías, Francesco Ricci, Iván Cantador. **Knowledge-based Music Retrieval for Places of Interest**. In: *Proceedings of the 2nd International Workshop on Music Information Retrieval with User-Centered and Multimodal Strategies (MIRUM 2012)*, *at the 20th ACM Conference on Multimedia (Multimedia 2012)*, pp. 19-24. ACM Press. ISBN 978-1-4503-1591-3.
- Ignacio Fernández-Tobías, Iván Cantador, Marius Kaminskas, Francesco Ricci. 2012. **Cross-domain Recommender Systems: A Survey of the State of the Art**. In: *Proceedings of the 2nd Spanish Conference on Information Retrieval (CERI 2012)*, pp. 177-189. Publicaciones de la Universitat Jaume I. ISBN 978-84-8021-860-32.
- Ignacio Fernández-Tobías, Marius Kaminskas, Iván Cantador, Francesco Ricci. 2011. **A Generic Semantic-based Framework for Cross-domain Recommendation**. In: *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011)*, *at the 5th ACM Conference on Recommender Systems (RecSys 2011)*. ACM Press, pp. 25-32. ISBN 978-1-4503-1027-7.

1.7 Structure of the document

The reminder of this document is structured as follows:

- In Chapter 2 we provide an overview of related work. Specifically, we review previous research on semantic relatedness metrics –distinguishing statistical and topological semantic similarities–, and briefly describe the main types of recommender systems –focusing on those that are more related to our approach, namely knowledge-based, context-aware, and cross-domain recommender systems.
- In Chapter 3 we present the proposed semantic-based framework for cross-domain recommendation, instantiated for the case study of suggesting musicians related to places of interest. In each section of the chapter, we describe a particular component of the framework. Specifically, we describe the used knowledge representation and extraction methods, and the investigated algorithms for computing cross-domain semantic relatedness and personalized recommendations.
- In Chapter 4 we present the experimental work conducted to validate our framework in the proposed case study. We discuss the results of several user studies aimed to address the research questions stated.
- Finally, in Chapter 5 we provide the conclusions of this work, and some directions for future research.

Chapter 2. Related work

Since the main functionality of our framework is to find and recommend those items in a target domain that are the most semantically related to a given item in a different source domain, in Section 2.1 we review representative work on semantic relatedness metrics. Next, in Section 2.2 we formulate the recommendation problem, and describe the main recommendation approaches proposed in the literature, with special emphasis on knowledge-based, context-aware, and cross-domain recommender systems, which are the approaches most related to ours.

2.1 Semantic relatedness across domains

An important research topic in the Natural Language Processing (NLP) area is to measure how related two different concepts are, that is, to compute a numeric score –typically a real value between -1 and +1– that indicates whether and to what extend the meanings of two concepts are somehow related. In this context, the proposed approaches in the literature can be classified into **statistical relatedness metrics**, which exploit the descriptions of the concepts at a lexical level, and **topological relatedness metrics**, which rely on the structure of the links between them.

Most of the initial research on semantic relatedness was focused on finding the degree of similarity between two terms in taxonomies and thesauri –mainly in WordNet (Miller, 1995). This fact is especially evident in the case of topological metrics, which are based on hierarchical semantic structures (Jiang & Conrath, 1997; Leacock & Chodorow, 1998; Lin, 1998; Rada et al., 1989; Resnik, 1995; Seco et al., 2004). In this context, three small, hand-crafted datasets (Finkelstein et al., 2002; Miller & Charles, 1991; Rubenstein & Goodenough, 1965), consisting of pairs of words together with their relatedness assessments, became the standard data sources for evaluation, and correlations between algorithmic outputs and human judgments were the preferred way to measure the performance of proposed semantic relatedness metrics.

The advent of Wikipedia brought the attention of researchers investigating on the topic due to its rich metadata, knowledge about multiple domains, and constant growth. Hence, Strube and Ponzetto (2006) recomputed some of the metrics proposed for WordNet on the Wikipedia's category graph. They showed that WordNet-based metrics achieved higher performance on small datasets, but were outperformed by Wikipedia-based metrics on large datasets. According to the authors, the poor performance of the WordNet-based metrics was due to the variety of word senses in the database, making the semantic disambiguation of terms difficult. In the quoted paper, the authors attempted to overcome such problem by selecting the first, most likely entries in Wikipedia's disambiguation pages. Afterwards, to the best of our knowledge, up to date the best performing approaches built upon Wikipedia have been Explicit Semantic Analysis, ESA (Gabrilovich & Markovitch, 2007), and Wikipedia Link-based Measure WLM (Milne & Witten, 2008).

We explain in detail the above semantic relatedness metrics in subsequent subsections. Specifically, in Sections 2.1.2 and 2.1.3 we review representative work on statistical and topological semantic relatedness metrics, respectively. Before that, in Section 2.1.1, we clarify the distinction between semantic relatedness and semantic similarity.

2.1.1 Definition of semantic relatedness

In several papers the terms "semantic similarity" and "semantic distance" have been used instead of "semantic relatedness" indistinctively. However, in most of the reviewed work a subtle distinction is made between such terms. Budanitsky and Hirst (2001) used the term **semantic similarity** when only predefined taxonomic relations (e.g. *is a, subclass of*) are taken into account, leaving the term

semantic relatedness for cases in which arbitrary relations are also considered. According to the authors, the semantic similarity between two concepts is determined by the likeness of their meanings (e.g. *teacher* and *school*), letting dissimilar concepts be related by other types of relations such as meronymy (e.g. *car* and *wheel*) and antonymy (e.g. *bright* and *dark*). Semantic relatedness is therefore a more general term than semantic similarity. Semantic distance, on the other hand, can simply be defined as the inverse metric of semantic relatedness, i.e., the higher the distance the lower the relatedness.

2.1.2 Statistical semantic relatedness

Statistical metrics measure the semantic relatedness between two concepts that are described as text documents in a corpus. They do it in terms of correlations at lexical level, by analyzing the co-occurrences of terms in the textual contents of the concepts' description documents.

Most of these techniques rely on the well-established Vector Space Model (VSM), a popular document representation model in the Information Retrieval area (Baeza-Yates & Ribeiro-Neto, 1999). In this model, a document d_i is represented as a vector $\overrightarrow{d_i} = (d_{i1}, \dots, d_{iN})$, where N is the size of the collection vocabulary –i.e., the number of distinct terms in the document corpus–, and the j-th component, d_{ij}, represents the relative importance of the term t_j in the document d_i. The semantic relatedness of two documents is then computed by means of the similarity between their vector representations.

In early work only binary features were used to describe the document vectors, being $d_{ij} = 1$ if $t_j \in d_i$, and 0 otherwise. In this case, the similarity between documents is computed in terms of the numbers of common and distinct terms. A concise review of these methods is presented in (Lee et al., 2005). For instance, the Common Features Model states that the relatedness is proportional to the number of common features:

$$\mathrm{rel}(d_i, d_j) = \frac{A}{A + B + C + D}$$

where A is the number of common terms to d_i and d_j; B and C are the numbers of terms that one of the two documents has, and the other has not; and D is the number of terms from the collection vocabulary that are not present in any of the two documents. Different variations of the previous formula lead to different metrics, like that of the Tversky's Ratio Model:

$$\mathrm{rel}(d_i, d_j) = \frac{A}{A + B + C}$$

in which only the terms that actually appear in any of the documents are taken into account; and that of the Distinctive Features model, which assumes that the document dissimilarity is greater if one document has terms the other does not have:

$$\mathrm{rel}(d_i, d_j) = \frac{A + D}{A + B + C + D}$$

A natural improvement of the binary models is to consider the term frequencies, $d_{ij} = \mathrm{freq}(d_i, t_j)$, i.e., the number of appearances of term t_j in document d_i. For this case, typical variations introduce an Inverse Document Frequency (IDF) to penalize the terms that appear in many documents, since they are less discriminative. The TF-IDF weighting schema (Baeza-Yates & Ribeiro-Neto, 1999) combines both ideas normalizing the frequency component:

$$d_{ij} = \frac{\mathrm{freq}(d_i, t_j)}{\max_k \mathrm{freq}(d_i, t_k)} \cdot \log \frac{N}{n(t_j)}$$

where $n(t_j)$ is the number of documents in which t_j appears, and N the number of documents in the corpus. Popular vector-based similarity metrics that often have been used with the TF-IDF weighting schema are Dice's and Jaccard's coefficients, the Overlap model, and specially the cosine model, which computes the angle between two vectors:

$$\mathrm{rel}(d_i, d_j) = \cos(\overrightarrow{d_i}, \overrightarrow{d_j}) = \frac{\overrightarrow{d_i} \cdot \overrightarrow{d_j}}{\|\overrightarrow{d_i}\| \cdot \|\overrightarrow{d_j}\|}$$

In general, the number of terms in the vocabulary is much greater than the number of documents in the corpus, and the term-document matrix –with the document vectors as columns– is very sparse. Because of this, Latent Semantic Analysis (LSA) (Deerwester et al., 1990) is usually performed. LSA is a dimensionality reduction technique that finds a low-rank approximation to the term-document matrix by performing a singular value decomposition. Specifically, the term-document matrix X is factorized as $X = U\Sigma V^T$, where U, V are orthogonal matrices, and Σ is a diagonal matrix of singular values. By retaining the k largest singular values, and their corresponding singular vectors, matrix approximations of rank k to the document vectors, $\overrightarrow{d_i^k}$ and $\overrightarrow{d_j^k}$, are obtained. This is equivalent to projecting the original documents into a k-dimensional subspace that captures the implicit semantics, and in which vector components no longer correspond to terms, but to non-trivial linear combinations of them. The relatedness between two documents is then computed by applying some vector similarity metric to their low-rank representations $\overrightarrow{d_i^k}$ and $\overrightarrow{d_j^k}$.

Lee et al. (2005) provided a comparison of the methods described above, evaluating their performance on manually built datasets with up to 364 documents. They found that LSA combined with a cosine similarity outperformed the rest of the methods, and that binary features models showed almost no correlation with human judgments.

Gabrilovich and Markovitch (2007) present an alternative to LSA called Explicit Semantic Analysis (ESA) that relies on Wikipedia to build the document vectors. In ESA, each article from Wikipedia represents a concept c_i, and is modeled as a term vector $\overrightarrow{c_i} = (c_{i1}, \dots, c_{iM})$ using the TF-IDF weighting schema, so that each term t_j has an associated relative importance c_{ij} for concept c_i, being M the number of distinct terms. Documents are also converted to term vectors $\overrightarrow{d_n} = (d_{n1}, \dots, d_{nM})$ using TF-IDF in the first place, and next these vectors are transformed into a concept space vectors $\overrightarrow{d'_n} = (d'_{n1}, \dots, d'_{nL})$ where L is the total number of concepts in Wikipedia, and each component

$$d'_{nl} = \sum_{m=1}^{M} c_{lm} \cdot d_{nm}$$

measures the importance of concept c_m to describe the document d_n. Finally, the semantic relatedness between two documents is computed as the cosine of their concept-based vectors. In contrast to LSA, each dimension has an easily interpretable meaning given by the corresponding Wikipedia article.

Summarizing, we highlight that the overall best performing statistical relatedness metric –in terms of reported correlation to human judgments– is ESA, followed by LSA –both of them using TF-IDF for concept weighting.

2.1.3 Topological semantic relatedness

Topological metrics measure the semantic relatedness between two concepts by considering explicit connections of such concepts in an ontology or a semantic network. These semantic link structures are used to estimate semantic relatedness e.g. by counting the length of the connection path between two given concepts, and considering that shorter paths indicate higher relatedness than larger paths.

In contrast to the statistical metrics, which rely on implicit semantic relations based on term co-occurrences, topological metrics make use of the explicit topologies of semantic networks existing e.g. in ontologies and thesauri like WordNet. These metrics can be further classified into two different categories according to the type of data they exploit, namely node-based and edge-based metrics.

Node-based metrics rely on the notion of information content (IC) to compute to what extent two concepts have information in common. Assuming the nodes and their links form a hierarchy, Resnik (1995) computes the relatedness between two concepts by taking their immediate subsumer node as the one that captures the above common information. The semantic relatedness is then computed as the information content of the least common subsumer (LCS) as follows:

$$\mathrm{rel}(c_1, c_2) = \mathrm{IC}\big(LCS_{1,2}\big) \triangleq -\log P\big(LCS_{1,2}\big)$$

where $P(c)$ is the probability of finding a concept c in the text corpus. Intuitively, the higher the probability, the more common the subsumer is, and the less information it carries. In (Resnik, 1995), that probability is estimated as the relative frequency of a concept c:

$$\hat{P}(c) = \frac{\mathrm{freq}(c)}{M}$$

where M is the total number of concepts identified in the collection.

Other metrics have been proposed based on Resnik's notion of information content. Lin (1998) presents a normalized version of the Resnik's metric that takes into account the IC of the involved concepts c_1 and c_2:

$$\mathrm{rel}(c_1, c_2) = \frac{2\, IC(LCS_{1,2})}{IC(c_1) + IC(c_2)}$$

Seco et al. (2004) also rely on the notion of information content, but propose an alternative definition that further exploits the taxonomic structure. They argue that a concept with more hyponyms requires more differentiation because it is less informative, in contrast to the most specific nodes with no hyponyms (i.e., leave nodes in the hierarchy), which express maximum information. In this case, the information content is computed as:

$$IC(c) = 1 - \frac{\log(\mathrm{hypo}(c) + 1)}{\log L}$$

where $\mathrm{hypo}(c)$ is the number of hyponyms of the concept c, and L is the total number of concepts in the taxonomy. The authors compare several node-based metrics using their IC definition and Resnik's IC probabilistic formulation. Although the authors report minor improvements, they claim that the advantage of their IC definition is that no analysis of the text corpus is required, and thus potential data sparsity problems are avoided.

Path-based metrics estimate the semantic relatedness between two concepts through their distance (number of links) in a semantic network. One of the first works in this direction was conducted by Rada et al. (1989), who compute the distance of the shortest path as:

$$\mathrm{rel}(c_1, c_2) = 2D - d(c_1, c_2)$$

where D is the maximum path length in the network, and $d(c_1, c_2)$ is the number of links in the shortest path connecting concepts c_1 and c_2. In the same line of work, Leacock and Chodorow (1998) propose a log-normalized version that also uses the shortest path:

$$\mathrm{rel}(c_1, c_2) = -\log \frac{d(c_1, c_2)}{2D}$$

A similar approach by Wu and Palmer (1994) considers the depth of the least common subsumer in the taxonomy in addition to the distance between the involved nodes:

$$rel(c_1, c_2) = \frac{2\, depth(LCS)}{d(c_1, LCS) + d(c_2, LCS) + 2\, depth(LCS)}$$

Of particular interest is the work done by Jiang and Conrath (1997), which presents a hybrid model that considers both the IC and the path between two concepts. The authors analyze some of the shortcomings of both strategies, namely the loss of accuracy that path-based approaches suffer when applied to broad domains with different types of links, and their dependence on the predefined hierarchy structure, which they argue as subjective. IC-based methods, on the other hand, are reported to have difficulties differentiating similarities to concepts from the same sub-hierarchy if their LCS does not change. Furthermore, according to Richardson and Smeaton (1995), polysemous terms have misleading IC values if their meanings are not considered, and only word frequencies are used. In order to address the first issue, Jiang and Conrath define the notion of link strength as the conditional IC of finding a concept c given its parent p:

$$LS(c, p) = -\log P(c|p) = -\log \frac{P(c \cap p)}{P(p)} = -\log P(c) + \log P(p) = IC(c) - IC(p)$$

where the third equality holds because c is a child of p, and thus $P(c \cap p) = P(c)$. This definition is further extended to take into account other factors such as node depth, link density, and link type. The weight of a link from a child node to its parent is finally:

$$w(c, p) = LS(c, p) \left(\frac{d(p) + 1}{d(p)}\right)^{\alpha} \left(\beta + (1 - \beta)\frac{\bar{E}}{E(p)}\right) T(c, p)$$

In this formula, $E(p)$ is the child count for the parent node p, $\bar{E}$ is the average child count in the whole hierarchy, $d(p)$ is the depth of p, and $T(c, p)$ is the prior weight for the link type between c and p. The parameters α and β control the influence of the depth and density terms, respectively. Instead of counting the number of edges between the nodes, the distance between two concepts is computed as the sum of the edge weights in the shortest path between them, excluding their LCS. In the particular case in which only the link strength is used to determine the weight ($\alpha = 0, \beta = 1, T(c, p) = 1$), the distance becomes

$$d(c_1, c_2) = IC(c_1) + IC(c_2) - 2IC(LCS)$$

and the final relatedness value $rel(c_1, c_2)$ can be obtained as an inverse function of the distance.

More recently, research focus has moved towards using new large-scale datasets with free link structure, such as Wikipedia, instead of using fixed hierarchical taxonomies and thesauri, such as WordNet. Milne and Witten (2008) propose a path-based method, called Wikipedia Link-based Measure (WLM), which uses the inner hyperlinks between Wikipedia articles describing concepts. The semantic relatedness of two concepts (articles) is estimated averaging scores for their incoming and outgoing links. For the outgoing links, the relatedness is computed as the cosine similarity between the outgoing link vectors of the two articles. Thus, two articles are similar if they point to the same documents with relevant links. In this context, the weight of a particular link (a component in the link vector) from article s to article t is given by:

$$w(s \to t) = \log \frac{|W|}{|T|}$$

where $|W|$ is the number of articles in Wikipedia, and $|T|$ is the number of articles that link to t. In the case that there is no link from s to t, the weight is 0. This formula is analogous to the IDF heuristic: the more an article is referred to, the less significant an individual link is. For the incoming links, the score is computed as:

$$\mathrm{rel}_{\mathrm{in}}(c_1, c_2) = \frac{\log(\max(|D_1|, |D_2|)) - \log|D_1 \cap D_2|}{\log|W| - \log(\min(|D_1|, |D_2|))}$$

Where D_1 and D_2 are the sets of articles that link to c_1 and c_2, respectively. Although the reported correlation with human judgments does not improve the best performing ESA, the authors claim that a competitive accuracy is obtained at a much cheaper cost, since the textual content of the articles can be mostly ignored.

Finally, Passant (2010) introduces the Linked Data Semantic Distance (LDSD), which also relies on arbitrary links, but is applied on Linked Data repositories instead of on Wikipedia. Concepts are therefore mapped to semantic resources –identified by URIs– instead of Wikipedia articles. The distance between two semantic resources is computed taking into account their direct and indirect links –both incoming and outgoing. In this case, the indirect links are restricted to a maximum of two edges, that is, through just one intermediate node. Different variants of the LDSD are described in that paper, being the most generic one as follows:

$$\begin{aligned} rel(c_1, c_2) &= \mathrm{LDSD}(c_1, c_2) \\ &= \left[1 + \sum_i \frac{N_d(l_i, c_1, c_2)}{1 + \log N_d(l_i, c_1, \cdot)} + \sum_i \frac{N_d(l_i, c_2, c_1)}{1 + \log N_d(l_i, c_2, \cdot)} + \sum_i \frac{N_{ii}(l_i, c_1, c_2)}{1 + \log N_{ii}(l_i, c_1, \cdot)} \right. \\ &\quad \left. + \sum_i \frac{N_{io}(l_i, c_1, c_2)}{1 + \log N_{io}(l_i, c_1, \cdot)} \right]^{-1} \end{aligned}$$

where $N_d(l, a, b)$ is 1 if there is a link of type l from concept a to concept b, and 0 otherwise.; and, similarly, $N_{ii}(l, a, b)$ is 1 if there is a concept that links to both concepts a and b, and $N_{io}(l, a, b)$ is 1 if there is a concept linked by both concepts a and b. From the formula, it can be observed that only paths of the forms $c_1 \to c_2, c_2 \to c_1, c_1 \to X \leftarrow c_2$, and $c_1 \leftarrow X \to c_2$ are taken into account.

2.2 Recommender systems

Recommender systems are a type of information filtering systems whose goal is to find those items –e.g. movies, music compositions, books, and news articles– most valuable for the users based on their preferences –tastes and interests. In contrast to search engines, recommender systems do not require the users to explicitly formulate their information needs as queries, and use the available information of the users' past preferences to proactively select valuable items.

Formally, the recommendation problem (Adomavicius & Tuzhilin, 2005) can be formulated as follows. Let $\mathcal{U}$ be a set of users, and $\mathcal{I}$ be a set of items. Let $g(u, i): \mathcal{U} \times \mathcal{I} \to \mathcal{R}$ be a function that measures the utility of an item $i \in \mathcal{I}$ for certain user $u \in \mathcal{U}$, where $\mathcal{R}$ is a totally ordered set, typically formed by numeric values called *ratings*, e.g. [1, 5] being 1 and 5 associated to the lowest and highest rating values, respectively. The goal of a recommender system is to find an item i_u^* such that

$$i_u^* = \mathrm{argmax}_{i \in \mathcal{I}}\, g(u, i)$$

for the target user u.

In general, the function g cannot be defined over the whole space $\mathcal{U} \times \mathcal{I}$. That is, the system does not have complete preference information of every user for every item, and therefore has to

somehow infer the missing preference information using the available past user preference evidence.

Numerous recommender systems have been proposed in the literature, most of which can be classified into one of the following categories:

- *Content-based recommender systems*, which estimate the utility of a target item as the "similarity" of the item and those items consumed by the user in the past.
- *Collaborative filtering systems*, which estimates the utility of a target item using the preferences that like-minded users expressed for that item in the past.
- *Hybrid recommender systems*, which combine content-based and collaborative filtering approaches.

Content-based (CB) recommender systems (Lops et al., 2011) represent items by means of attributes that describe item characteristics, and numeric weights that measure the relative importance of such attributes for the items' description. Hence, as in the Vector Space Model, item profiles $\vec{\imath}$ are represented as vectors whose components correspond to the items' attribute weights. Likewise, user profiles $\vec{u}$ are typically defined by aggregating the vectors of the items the users liked in the past. Then, the utility of an item i for a user u is heuristically computed using some vector similarity metric, e.g. the cosine similarity $g(u,i) = \cos(\vec{u},\vec{\imath})$. One of the problems of content-based recommendation approaches is that information about item attributes is required, which may result in a *restricted content analysis*. Also, since these approaches only exploit the target user's preferences, their generated recommendations may suffer from *content over-specialization* and *lack of diversity*. Furthermore, in case the user has no (or a few) preferences, a content-based approach cannot suggest the user with (valuable) items, fact that is known as the *cold-start* problem.

Collaborative filtering (CF) recommender systems (Herlocker et al., 1999; Koren & Bell, 2011) do not require any knowledge about item characteristics, and only rely on user feedback to compute recommendations. User-based collaborative filtering (Desrosiers & Karypis, 2011) –also known as k nearest neighbors filtering, kNN– identify a set $N(u)$ of users who expressed preferences similar to those of the target user u, and the final utility of an item for u is computed by aggregating the preferences for that item of the similar-minded users $N(u)$. Even though no information about the items is needed, a significant number of users have to evaluate the items in order to be eligible for recommendation. This is known as the *sparsity* problem. Related to this, and similarly to content-based recommendation approaches, collaborative filtering suffers from the *cold-start* problem, in the sense that an item without ratings cannot be recommended, or a user without (enough) ratings cannot receive (valuable) item recommendations.

Hybrid recommender systems (Burke, 2002) combine both content-based and collaborative filtering approaches to compensate their particular limitations. The combination of recommender systems can be performed in different ways, e.g. by aggregating the systems' recommendation scores, in cascade by filtering out items that are sequentially non recommended by the systems, by integrating content-based item characteristics into collaborative filtering models, or by using a switch strategy in which only one system is chosen as active depending on input data.

The area of RS has gained much attention in the last years from both the research community and the industry. Due to the growing number of heterogeneous sources of information, especially in the Web, **knowledge-based recommender systems** –which exploit explicit knowledge about users and items–, are a compelling approach for addressing novel recommendation applications, such as **context-aware recommender systems** –which in addition to user preferences consider contextual signals like the time, and the user's current location and mood–, and **cross-domain**

recommender systems –which exploit data in a source domain to provide recommendations in a different, target domain. In subsequent subsections we review representative work on each of these types of recommendation approaches.

2.2.1 Knowledge-based recommender systems

Despite their success in a large number of applications, content-based and collaborative filtering recommendation approaches may not be the most suitable solution in cases when a detailed understanding of the users' preferences is needed, e.g. recommendations of restaurants (Burke, 2000), hotels (Jannach et al., 2009), and financial products (Felfernig et al., 2007). In these cases, the number of available ratings is usually too low, and item-oriented ratings do not provide information about the user's fine-grained preferences on particular item characteristics, e.g. the cuisine type and menu average price of a restaurant, the location and transport facilities of a hotel, and the costs and potential risks of a financial product.

Knowledge-based recommender systems aim to address the above problems by exploiting explicit user preferences on item characteristics, and (semantic) information of the application domain (Felfernig et al., 2011). This lets tackle the cold-start problem of CB and CF approaches. However, the exploited knowledge has to be extracted and formalized into executable representations that machines can process, a problem which is often referred to as the *knowledge acquisition problem*.

User annotations in social tagging systems are a type of explicit metadata that has been exploited for recommendation tasks. In these systems, users can assign freely chosen terms (tags) to the items in order to provide them with personal descriptions. The whole set of tags forms a collaborative (semi-structured) knowledge representation known as *folksonomy* (Vander Wal, 2007). Upon such knowledge representation, a user's preferences can be modeled with the tags the user assigns to items, and an item's description can be built by aggregating the tags that the item is assigned by users. In (Cantador et al., 2010) different methods are proposed for building tag-based user and item profiles, and adapt the well-known vector space and BM25 information retrieval models (Baeza-Yates & Ribeiro-Neto, 1999) to compute content-based recommendations. In (Hotho et al., 2006) the authors present *FolkRank*, an adaptation of the well-known PageRank algorithm (Brin & Page, 1998) for searching and ranking in folksonomies. In this approach, users, tags, and items are represented as nodes in an undirected graph over which a weight propagation strategy is performed. Personalized recommendations are obtained by considering a user preference vector in the propagation strategy, so that the graph nodes' scores are computed also taking the user's tastes into account.

In a different direction, some works make use of semantic-based knowledge sources to enhance user and item profiles. In (Mobasher et al., 2003) an alternative to item-based collaborative filtering is proposed, in which semantic similarities of items are integrated with rating-based similarities. In (Middleton et al., 2004) user profiles are modeled with ontologies for recommending research papers. Both user preferences and research papers are classified using an ontology of research topics, and papers bookmarked by people with interests similar to the target user are recommended. In (Cantador et al., 2011) a semantic layer is built using ontological knowledge to connect the (concept-based) user and item spaces. User and item profiles are extended through semantic relations using a constrained spreading method, and context-aware recommendations are generated considering the set of concepts that have been involved in the user's recent actions over items.

Finally, constraint-based recommender systems exploit predefined knowledge sources that contain explicit rules about how to relate user preferences –expressed as needs or requirements– with item features (Felfernig et al., 2011). A knowledge base is usually defined by two components.

The first component is the customer and product tuples, (V_C, V_{PROD}), which describe the possible user requirements V_C for the different item feature values V_{PROD}. A particular user's requirement on certain item feature is an instantiation of those tuples, e.g. *preferred_genre=comedy*, and *maximum_duration=120* in the movie domain. The second component consists of sets of constraints, (C_R, C_F, C_{PROD}), where C_R and C_{PROD} are restrictions on the possible instantiations of customer requirements and product features, and C_F are the rules that relate customer requirements with product properties, e.g. users that do not like pasta should not be recommended Italian restaurants. In this context, the recommendation task is formulated as a constraint satisfaction problem over $C_R \cup C_F \cup C_{PROD}$, and a consistent recommendation is an assignment of the variables in V_C and V_{PROD} that does not violate any established constraint.. Besides the building of a knowledge base, these types of RS have to address the design of user interfaces to guide and support the user during the (interactive) recommendation process. In (Felfernig & Shchekotykhin, 2006) sets of possible user dialogs and transitions are defined using finite state models, and in (Mahmood & Ricci, 2007) users select which item features they are interested in using forms.

In this thesis, we exploit semantic-based knowledge sources to build networks that relate different domains. These networks are automatically built with classes, instances and properties existing in ontologies. A weight spreading algorithm is used to filter the relevant instances for recommendation. Moreover, in one of the proposed personalized recommendation methods (see Section 3.4.2), user preferences are modeled with ontological concepts, so that their prior relevance is propagated in the semantic networks.

2.2.2 Context-aware recommender systems

The design and implementation of a recommender system highly depends on the application it targets, and thus, in some cases, information about users and items is not enough to provide useful item recommendations. For example, the user's current location is crucial to suggest relevant restaurants to go, the time when the user expressed her movie preferences can be determinant for deciding which movie to be recommended and watch at present, and the user's current mood can influence how recommended music compositions are perceived when listened. In all these cases, contextual information must be taken into account for providing meaningful recommendations under particular circumstances.

Quoting Dey (2001), "context is any information that can be used to characterize the situation of an entity." In information retrieval and filtering systems, an entity can be a user, an (information) item, or an experience the user is evaluating (Baltrunas & Ricci, 2013), and any signal –such as device, location, time, social companion, and mood– regarding the situation in which a user interacts with an item can be considered as context.

The goal of context-aware recommender systems (CARS) is to improve the quality of item recommendations by exploiting information about the context C in which user preference predictions for such item are made. This can be stated by reformulating the recommendation problem with an extended utility function g from the two-dimensional space $\mathcal{U} \times \mathcal{I}$ to a multidimensional space (Adomavicius & Tuzhilin, 2011) as follows:

$$g(u, i. C): \mathcal{U} \times \mathcal{I} \times \mathcal{C}_1 \times \ldots \times \mathcal{C}_K \rightarrow \mathcal{R}$$

where the $\mathcal{C}_j$ are the contextual signals –e.g. location, time, mood, and social companion– considered in the recommendation process.

Context-aware recommendation approaches are usually classified into three categories (Adomavicius et al., 2005), namely *contextual pre-filtering* (Baltrunas & Ricci, 2009), *contextual post-filtering* (Panniello et al., 2009), and *contextual modeling* (Karatzoglou et al., 2010; Oku et al., 2006;

Rendle et al., 2011) approaches. Contextual pre- and post-filtering approaches are based on context-unaware recommendation methods that are applied on preference data that are pre-processed, or are used to generate recommendations that are post-adjusted, in both cases according to the user's current context. Contextual modeling, on the contrary, extends the user-item preference relations with contextual information to compute recommendations.

One of the major difficulties in the development of CARS is the lack and sparsity of contextualized user preference data. Adomavicius and Tuzhilin (2011) identify three ways of obtaining such data: i) explicitly, by asking the users directly about their preferences under certain contexts, ii) implicitly, by extracting the contextual information from available data sensors, and iii) by automatically inferring the contextual information using data mining techniques.

In this thesis, we propose a method for linking concepts from different domains using explicit semantic relations. We implement and exploit the method for a particular context-aware recommendation task, suggesting musicians related to places of interest (see Chapter 3). We further investigate two methods for integrating user preferences and contextual (semantic) relatedness, namely personalized graph-based similarities and ranking aggregation. Both methods follow a pre-filtering contextualization approach, since only items that appear in the semantic networks are related to the context, and can thus be recommended.

2.2.3 Cross-domain recommender systems

Most of the developed recommender systems target items that belong to a single domain. For instance, Netflix suggests a user with movies and TV series, and Last.fm makes personalized recommendations of music artists and compositions. Nowadays, nonetheless, there are many leisure and e-commerce Web sites, like Amazon, which could take benefit from exploiting user preferences on diverse types of items to provide recommendations in different but somehow related domains. Recommendations across domains could mitigate the cold-start problem when little information about the user's preferences is available in a target domain, and are potentially more diverse and serendipitous than single-domain recommendations (Winoto & Tang, 2008). Moreover, recommendations in a particular domain could be computed based on a "context" that is expressed in a different but somehow related domain.

The goal of cross-domain recommender systems is to suggest items in a target domain by exploiting user preferences and/or domain knowledge available in a different source domain. Hence, a major challenge for making cross-domain recommendations is to establish methods for transferring knowledge across domains, in which usually there is little or no overlap between user preferences or item attributes.

As a fairly recent research topic, so far cross-domain recommendation has been mostly addressed in the collaborative filtering setting, where there is some user preference (rating) overlap between domains, and where item attributes are not needed. In a seminal paper, Winoto and Tang (2008) identify three issues to investigate in cross-domain recommendation: the existence of global correlations between user preferences for items in different domains, the method to exploit data in a source domain for predicting preferences on a target domain, and the methodology and metrics to evaluate cross-domain recommendations. In (Li, 2011) Li surveys works on cross-domain collaborative filtering. He classifies existing approaches according to the type of knowledge transferred, namely rating patterns, latent features, and user/item inter-domain correlations. Tiroshi and Kuflik (2012) evaluate the influence of the involved domains in the recommendation using a kNN approach in which the user's neighborhood in the target domain is selected among the most similar users from a source domain. To address the problem of little domain overlap, Cremonesi et

al. (2011) model user and item similarities through graphs in which all possible paths linking two users or items are used to enhance collaborative filtering algorithms.

Besides collaborative filtering, there have been some attempts to establish semantic relations between items of different domains. In (Loizou, 2009) Loizou annotates and links items by means of concepts and relations extracted from Wikipedia. Then, using such relations, users and items are incorporated into a graph upon which a probabilistic recommendation model is built. Social tags have also been used to establish relations between items of different domains. In (Kaminskas & Ricci, 2011) Kaminskas and Ricci show that emotional tags can be used to effectively select music that fits places of interest. Shi et al. (2011) utilize tags to build inter-domain user-to-user and item-to-item similarities. These similarities are proportional to the numbers of tags shared by profiles from different domains, and are incorporated as constraints into a probabilistic matrix factorization model for collaborative filtering.

In this thesis, we implement the proposed framework for automatically building cross-domain semantic networks with structured information extracted from Linked Data repositories. These networks connect concepts from several domains through semantic properties, establishing thus bridges that can be used to support or perform cross-domain recommendation.

Chapter 3. Proposed approach

In this chapter we present our semantic-based framework to support cross-domain recommendations. We instantiate and describe it for the particular case study of recommending music artists and compositions (semantically) related to places of interest, which we introduced in Section 1.4. In the subsequent sections we explain each of the components of the framework. Specifically, in Section 3.1 we describe our model for cross-domain knowledge representation, and in Section 3.2 we detail the developed data acquisition process. Next, in Sections 3.3 and 3.4 we respectively present the semantic relatedness metrics and recommendation algorithms performed over the semantic networks of the framework.

3.1 Cross-domain knowledge representation

Our generic framework for cross-domain recommendation is built upon an ontology-based knowledge representation, namely a graph/network of semantic entities from different domains interlinked by ontology properties.

Entities can be either classes or instances. Classes are types of concepts, such as 'city', and instances are particular individuals of classes, such as 'Madrid', as an instance of the class city. Moreover, semantic relations can link classes (e.g. a city 'belongs to' a country, being country a class), instances (e.g. Madrid 'is the capital of' Spain, being Spain an instance of the class country), or both types of entities (e.g. Madrid 'is a' city). Links can express hierarchical relationships, e.g. 'subclass of' and 'instance of', or have arbitrary meanings.

With the proposed semantic network-based representation, the goal is to automatically find and score paths between an instance in a source domain (e.g. a POI) to instances in a target domain (e.g. musicians), and select (recommend) the reached target instances with the highest scores. To address this problem we exploit DBpedia, the major Web-based multi-domain knowledge base, which can be queried to easily gather structured semantic data.

In the following we explain how we define semantic *class networks* across domains, and how we instantiate such class networks as *instance networks* with data automatically obtained from the DBpedia ontology.

3.1.1 Class network

The first stage of our approach comprises the definition of a semantic network that links/relates a limited number of classes belonging to several domains, including the source and target domains of interest. The classes, together with the properties that link them, could be obtained from one or more ontologies, such as those existing in Linked Data. As a proof of concept and preliminary implementation of our framework, we restrict our focus to the subspace of the DBpedia ontology composed of classes that belong to the two domains of interest of our case study, i.e., POIs and musicians, and properties existing between instances of such classes.

More specifically, we establish a network consisting of a directed acyclic graph (DAG) whose nodes represent the identified classes, and whose edges represent the selected relations (Figure 2). In this graph there is a target node that has not out edges, and corresponds to the class whose instances will be recommended ('Musician' in Figure 2).

The selection of the relevant classes and relations is guided by domain experts. For the addressed case study, Figure 2 shows the cross-domain graph linking musicians and places of interest. In the figure, 'POI' and 'Musician' classes represent the starting and ending nodes in the

class network. Analyzing DBpedia, we have identified three types of potentially useful semantic paths from POIs to musicians:

- **Location paths**. A particular POI may be linked to musicians who were born, died or lived in the city of the POI. For instance, Arnold Schoenberg was born in Vienna, which is the city where Vienna State Opera is located.
- **Time paths**. A POI may be linked to musicians who were born, died or lived in the same time period (e.g. year, decade, century) in which the POI was built or opened. For instance, Gustav Mahler was born in 1869, the same decade when Vienna State Opera was built.
- **Category paths**. A POI may have Architecture categories (e.g. architectural styles and eras, building types) that are related to Music categories (e.g. music genres and eras, musician types), through explicit relations with History and Art categories. In this way, a musician having Music categories related to architecture categories of the input POI may be related to the POI as well. For instance, 'Wolfgang A. Mozart' was a classical music composer, and classical compositions are played in Opera houses, which is the building type of the 'Vienna State Opera'; 'Antonio Flores' was a famous Spanish Flamenco singer, Flamenco is a Romani music genre widely diffused in Andalusia, which is the Spanish region most influenced by the Moorish architecture, and the Moorish Revival architecture is the architectonical style of 'Las Ventas', the main bullring in Madrid.

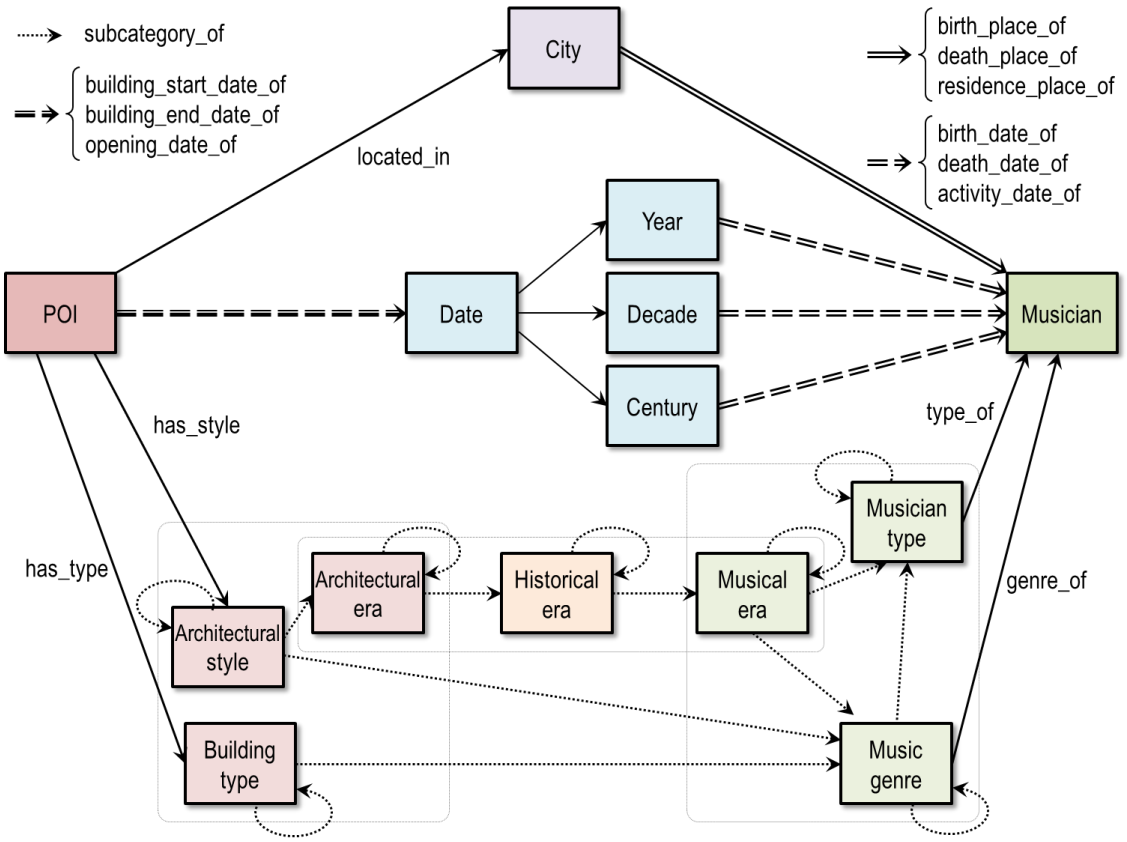

Figure 2. Class network used in the framework for the addressed POI-to-musician recommendation case study; some of the architecture, history and music concepts group various DBpedia classes.

Other arbitrary semantic relations between POIs and musicians could be taken into account, e.g. direct relations such as 'Gustav Mahler was the director of Vienna State Opera', and complex non-directed relations such as 'Ana Belén (a famous Spanish singer) composed a song whose lyrics are about La Puerta de Alcalá (a well-known POI in Madrid, Spain).' However, in this thesis we do not consider these relations since they are not captured by explicit DBpedia ontology properties. We plan to automatically identify and extract such relations from Web documents in future work, as discussed in Chapter 5.

In our framework we assign relevance values to the considered classes and instances, which can be used for recommendation. These values may be assigned by domain experts, or could be obtained from user profiles. For instance, a domain expert may assign higher relevance to the class 'City' than to the class 'Building type', since the former may be considered more informative to link a POI with related musicians. Similarly, classes like 'Opera composers' and 'Classical music' may receive high relevance if the user has a preference for them, hence producing personalized associations. Moreover, we consider the case in which relations also receive relevance values, e.g. to measure the strength with which 'Art Deco' architectural style and 'Swing' music genre (both emerged in the 1930s) with respect to other more/less related categories.

The model shown in Figure 3 captures the above discussion. Formally, let $\mathcal{E} = C \cup I$ be the set of class and instance entities. We assume a function $\mathrm{rel}_e: \mathcal{E} \rightarrow \mathbb{R}_+$, with $\mathbb{R}_+ = \{x \in \mathbb{R} : x \geq 0\}$ representing the relevance value assigned to entities $e \in \mathcal{E}$. We also assume a function $\mathrm{rel}_r: \mathcal{R} \rightarrow \mathbb{R}_+$ representing the relevance value assigned to relations $r \in \mathcal{R} \subseteq \mathcal{E} \times \mathcal{E}$ between pairs of entities. Given these assignments we aim to compute a score (see subsequent sections) that measures to what extent an instance is related to another one when they are not directly connected by a relation. In the figure, $C_A, C_B \in C$ are classes –types of concepts– such as 'City' and 'Musician', whereas $\mathrm{I_A}, \mathrm{I_B} \in I$ represent arbitrary instances of such classes. The dashed lines represent 'is a' relations between classes and instances, e.g. the instance 'Vienna' is a 'City', or 'Gustav Mahler' is a 'Musician'. We note that relevance values of entities can be specified both for classes and instances, and likewise relevance values for relations between classes and also between instances.

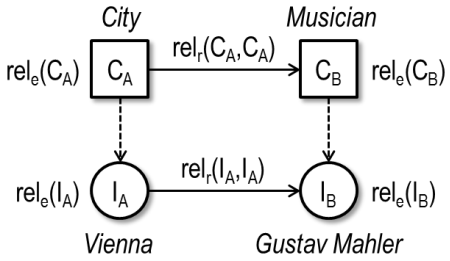

Figure 3. Relevance values assigned to semantic entities (classes and instances) and relations between them.

For simplicity purposes, in our initial framework implementation and evaluation, we set all class relevance values to 1, and manually established the relevance values of the class relations (1 for all relations except 0.25 for the residence place relation, 0.25 and 0.5 for the century and decade date relations, and 0.5 for the architecture and music subcategory relations). The relevance values of the instances and their relations were established by TF-IDF weighting with the entity/relation occurrences in the graphs of our POI collection. In future work we shall investigate and evaluate alternative strategies to automatically establish the distinct relevance values; see Chapter 5 for more details.

3.1.2 Instance network

In the second stage of our approach, from the established class network, we automatically build a second network that links a given instance in the source domain to related instances in the reminder and target domains. This is obtained by accessing the used structured data repositories (i.e., DBpedia) for instance data associated to the classes in the class network. Figure 4 shows a part of the instance sub-network that links the POI 'Vienna State Opera' and the musician 'Gustav Mahler.' The full instance network of a POI is obtained by aggregating all the instance sub-networks linking the POI to semantically related musicians.

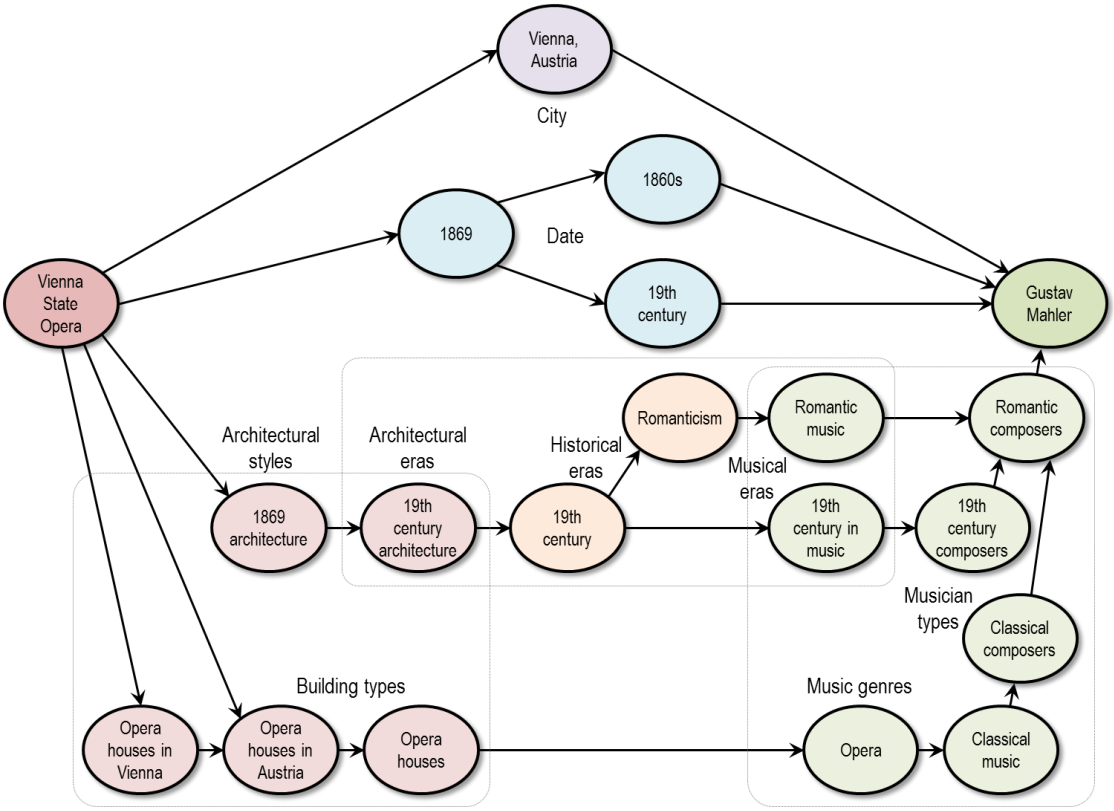

Figure 4. An example of instance sub-network linking the POI Vienna State Opera with the composer Gustav Mahler.

An instance network has weights assigned to the relations between pairs of instances. These weights are computed from the relevance values of the relations linking the instances and their classes. Specifically, we define the weight of a relation between two instances as a function w:I × I ⟶ $\mathbb{R}_+$ that depends on the relevance values of the properties between the two instances $I, I' \in$ I, and between their classes $C_I, C_{I'} \in$ C:

$$w(I, I') = f\big(\mathrm{rel}_r(I, I'),\ \mathrm{rel}_r(C_I, C_{I'})\big)$$

where $w(I, I') = 0$ if there is no link from I to I'.

In the experiments presented in this thesis, f is a linear combination of the relevance values assigned to the semantic properties between I and I', and between C_I and $C_{I'}$:

$$w(I, I') = \lambda \cdot \mathrm{rel}_r(I, I') + (1 - \lambda) \cdot \mathrm{rel}_r(C_I, C_{I'}), \lambda \in [0,1]$$

For simplicity purposes, the parameter λ was set to 0.5 in our experiments in order to give the same importance to relations between classes and relations between instances.

3.2 Cross-domain knowledge extraction

In this section, we present the methods we have developed to acquire knowledge from DBpedia for building the class and instance networks described in Section 3.1. The implementation of the proposed methods has been specialized for the POI-to-musician recommendation case study. The implementation is, however, modular and flexible, and thus can be easily adapted to other domains of interest.

The methods operate in both offline and online mode. In offline mode they iteratively query DBpedia to obtain all the entities (classes and instances) and relations considered in the framework for the source and target domains. The acquired data is then stored into a database. In online mode (i.e., at execution time) and for a particular input POI, we access the database to retrieve instances and relations related to the input, and build the corresponding POI-focused cross-domain semantic network. Over this semantic network, graph-based retrieval algorithms return ranked lists of target instances.

In the subsequent subsections we detail each of the above methods. Specifically, in Sections 3.2.1 and 3.2.2 we describe the developed methods to build and link domain taxonomies that will let categorize and relate classes and instances from the source and target domains. In Section 3.2.3 we describe a method to acquire particular domain knowledge about the considered classes and instances.

3.2.1 Building domain taxonomies

The developed implementation of our framework starts with a method for building domain taxonomies that let identify and relate instances from the source and target domains of interest. Since Wikipedia is a source of universal knowledge that spans a great number of domains, the taxonomies are built using Wikipedia categories. These taxonomies correspond to the instances of the 'category paths' in our framework (see the example in Figure 4).

In Wikipedia, and consequently in DBpedia, a concept has a number of categories. These categories are manually established by Wikipedia administrators following a number of naming conventions and categorization rules[9]. The categories share hierarchical relations among them, forming a very large semantic graph that links categories in many different domains, most of which do not belong to the relevant domains. Figure 6 shows a Wikipedia sub-graph that links categories belonging to the Architecture and Music domains. In the figure we highlight those edges that link the above domains of interest. We note that there are other edges that link such categories with others, but these belong to non-relevant domains. Our goal is to identify the relevant entities and relations, taking into account the complexity of the category graph in Wikipedia.

To select only those categories belonging to a particular domain of interest (e.g. Architecture, Music), and use these categories for building associated domain taxonomies, we have developed the algorithm depicted in Figure 5.

[9] Wikipedia's categorization, http://en.wikipedia.org/wiki/Wikipedia:FAQ/Categorization

```
1.  global input:
2.     rootCategories: root categories of the taxonomies
3.     allowedPatterns: allowed category name patterns for expansion
4.     forbiddenPatterns: forbidden category name patterns
5.     maximumLevel: maximum level/depth of a taxonomy
6.  global output:
7.     taxonomies: taxonomy index
8.  procedure createTaxonomies
9.     taxonomies = createTaxonomyIndex()
10.    foreach root in rootCategories
11.       taxonomies.add(root, null, 0)
12.       expandCategory(root, 0)
13. procedure expandCategory
14.    local input:
15.       parent: (parent) category to expand
16.       level: level of the parent category in the taxonomy
17.    if level <= maximumLevel
18.       children = DBpedia.getDirectSubcategories(parent)
19.       foreach child in children
20.          if isAllowedCategory(child, allowedPatterns)
21.             if not taxonomies.contains(child)
22.                taxonomies.add(child, parent, level + 1)
23.                expandCategory(child, level + 1)
24.             if not isForbiddenCategory(child, forbiddenPatterns)
25.                taxonomies.add(child, parent, level + 1)
```

Figure 5. Pseudocode of the domain taxonomy building algorithm.

The algorithm performs a Depth-first Search process over the Wikipedia's category graph. The process starts from a number of input "root" categories (line 2 in the pseudocode of Figure 5), which are located at the highest levels of the domain taxonomies. In Figure 6 root categories are in dark colors, e.g. 'Visitor attractions' and 'Architectural styles' for the architecture domain, and 'Music people' and 'Music genres' for the music domain.

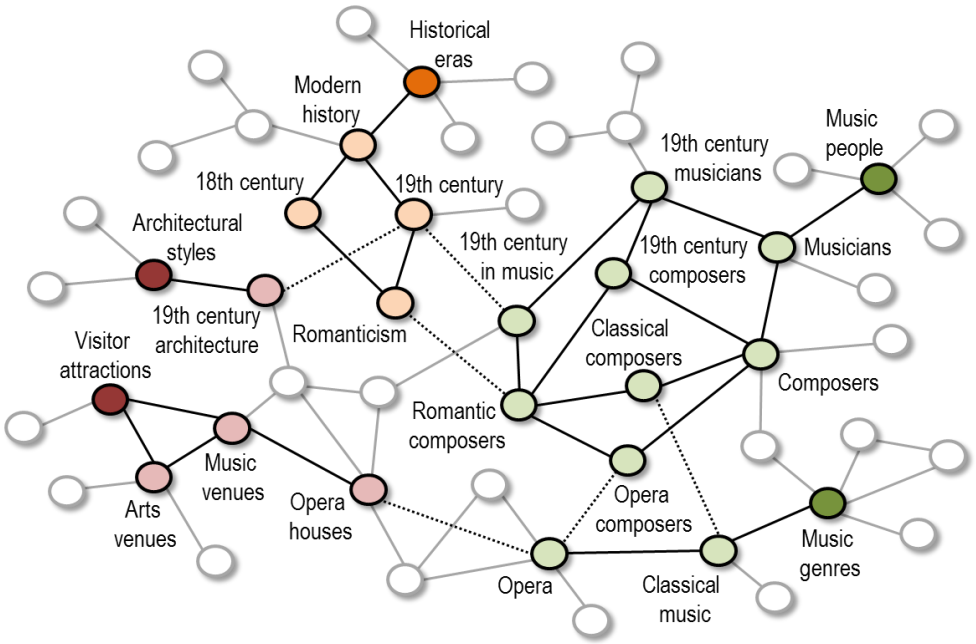

Figure 6. Example of a semantic sub-graph extracted from Wikipedia ontology linking categories of Architecture, History and Art, and Music. Relations between categories of different domain taxonomies are represented with dotted lines. Root categories of the taxonomies in each domain are highlighted in dark colors.

Initially, the root categories are added to the taxonomy index with null parents (line 11). Next, for each root category, we obtain its subcategories from DBpedia (line 18). The following is a RDQL query that retrieves the direct subcategories (linked through the property `skos:broader`) of the 'Architectural styles' root category:

```
SELECT ?subcategory WHERE {
?subcategory
<http://www.w3.org/2004/02/skos/core#broader>
<http://dbpedia.org/resource/Category:Architectural_styles> }
```

If a subcategory has an allowed name for expansion or has not a forbidden name (lines 20 and 24), it is added to the taxonomy having as parent the corresponding root category. If it has an allowed name for expansion, its subcategories are obtained from DBpedia and the search process is recursively repeated until it reaches a maximum taxonomy level (set to 10 in our implementation) or a category with no subcategories. The recursive process also stops when reaching a category that was already included in the taxonomy (line 21). Figure 7 shows small parts of the three taxonomies built for the Architecture, History and Art, and Music domains, which are constructed from a main category that corresponds to one of the input root categories for the algorithm. Each level going down in the taxonomies represents successively specialized categories, which are found following semantic properties that express hierarchical relationships. Links between categories of taxonomies from different domains are represented in dotted lines.

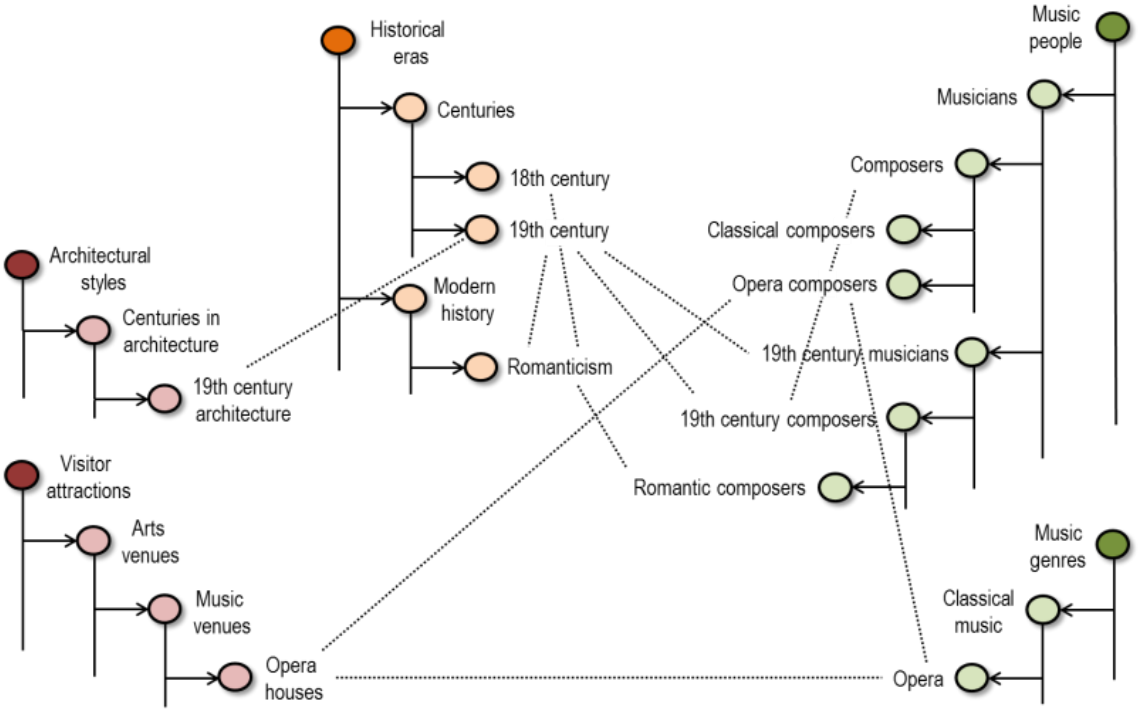

Figure 7. Parts of the interconnected taxonomies for the Architecture, History and Art, and Music domains, extracted from the Wikipedia categories graph.

The root categories as well as the allowed and forbidden category name patterns are established by a domain expert. Table 1 shows the root categories considered for building the architecture and music taxonomies in our case study:

Table 1. Root categories considered for the architecture and music taxonomies. The prefix **cat:** should be replaced by **http://dbpedia.org/resource/Category:**.

Root architecture categories	Root music categories
cat:Architectural_styles	cat:Music_genres
cat:Architectural_history	cat:Musical_eras
cat:Architecture_by_country	cat:Centuries_in_music
cat:Years_in_architecture	cat:Musicians_by_genre
cat:Establishments_by_century	cat:Composers_by_genre
cat:Visitor_attractions	cat:Singers_by_genre
	cat:Musical_groups_by_genre

Despite the naming conventions and categorization rules followed by Wikipedia administrators when assigning categories to concepts, the Wikipedia's category graph, which involves a large number of domains, and multiple categorizations within a particular domain, is complex and contains irregularities and mistakes. For instance, the graph is cyclic, in the sense that there are cases where B is a subcategory of A, C is a subcategory of B, and C is a parent category of A, which implies that A is a subcategory of B.

Our algorithm deals with the complexity of the category graph, and selects relevant categories that belong to a particular domain. The algorithm 1) only extends –i.e., visit subcategories of– categories whose names satisfy some pattern established by the domain expert –e.g. categories whose names end with the term 'architecture'; see line 20 in Figure 5–, 2) stops when reaching a category already visited –to avoid cycles; line 21–, and 3) does not incorporate into a taxonomy categories whose names satisfy a pattern declared as forbidden by the domain expert because it represents categories that are noisy or not valid for our categorization purposes –e.g. those categories whose names contain the term 'unfinished'; line 24–. Table 2 shows several examples of allowed category name patterns (to expand) and forbidden category names (to incorporate) considered for building the architecture and music taxonomies.

Table 2. Category patterns allowed to extend, and some patterns forbidden to incorporate in the taxonomy building process. An asterisk * means any sequence of characters.

Architecture taxonomy	
Allowed category name patterns	**Forbidden category name patterns**
*architecture	*architects*
*architectural_styles	*future*
establishments	*organizations*
venues	*unfinished*
Music taxonomy	
Allowed category name patterns	**Forbidden category name patterns**
*genres	*awards*
composers	*clubs*
musicians	*dancers*
singers	*magazines*

The built Architecture and Music taxonomies contained 16037 and 10255 categories, and were created with 25 and 53 forbidden category name patterns, respectively.

3.2.2 Linking domain taxonomies

Once the domain taxonomies are built, we establish semantic links between such taxonomies, as shown in Figure 7. We distinguish two types of category links, namely direct links from a category in the source domain to a category in the target domain, and non-direct links that relate categories in these domains through direct links to categories in a third related domain.

In our case study we identified Art and History as a third domain that can be used as a bridge to link the Architecture and Music domains. Figure 7 shows examples of direct and non-direct links between the above domains. For instance, 'Opera houses' is directly linked to 'Opera composers', and '19th century architecture' is linked to 'Romantic composers' through the '19th century' and 'Romanticism' historical era categories.

To find non-direct links in DBpedia we created the intermediary domain taxonomy modifying the algorithm explained in section 3.2.1 by considering an additional expansion criterion. This criterion states that a category is expanded if and only if, at some level, one of its subcategories belongs to any of the already built architecture and music taxonomies. We thus limit the scope of the history taxonomy to those History and Art categories that are somehow related to the Architecture and Music domains. In this case, the search process starts from the root category `http://dbpedia.org/resource/Category/Historical_eras`. The built Art-History taxonomy contained 5845 categories.

Once all the taxonomies are built, we automatically extract from their linked hierarchical structures semantic paths that relate Wikipedia categories in the source and target domains. We note that these paths may contain sequences of direct subcategories belonging to a single domain, e.g. '19th century' and 'Romanticism' history categories in a path that relates '19th century architecture' with 'Romantic composers.'

In the built taxonomies, we extracted 1086 direct paths and 11458 non-direct paths between architecture and music categories. These paths linked 1596 architecture categories (9.9% of the architecture categories) with 615 music categories (6.0% of the music taxonomy), without considering the categories linked through hierarchical relations within each taxonomy.

3.2.3 Acquiring domain knowledge

The next task of the framework instantiation process consists of acquiring source and target domain instances (i.e., POIs and musicians) and metadata for linking such instances.

As done in the majority of the Linked Data repositories, in DBpedia data is stored and retrieved in the form of triples, which are composed of subject-predicate-object elements, such as `<Arnold_Schoenberg, was_born_in, Vienna>`. To obtain the POIs of a particular city, we first launch a SPARQL query to DBpedia for retrieving all the triples whose object is the city. In other words, we query DBpedia for all the entities (subjects) that are related (through some semantic property) to the city's DBpedia resource, e.g.:

```
SELECT ?subject ?property WHERE {
?subject ?property <http://dbpedia.org/resource/Vienna> }
```

The retrieved (subject) entities are not necessarily places. To identify those subjects that are POIs we query DBpedia again to obtain all the triples of such entities, e.g.:

```
SELECT ?property ?object WHERE {
<http://dbpedia.org/resource/Vienna_State_Opera> ?property ?object }
```

We thus get all the metadata (properties and objects) associated to the entities, such as `dbpprop:yearBuilt` and `1869`, among others. Some of these metadata will be stored and exploited in our framework, as explained later. A retrieved (subject) entity is considered as a POI if at least one of its triples has a property existing in a predefined list of properties that link to categories in the architecture taxonomies (Table 3), for example `dbpprop:architecture` and `cat:Opera_Houses`.

Table 3. Properties used to obtain POIs/musicians linked to architecture/music categories. The prefix **dbpprop:** should be replaced by **http://dbpedia.org/property/**, the prefix **dbpedia-owl:** by **http://dbpedia.org/ontology/**, and the prefix **dcterms:** by **http://purl.org/dc/terms/**.

POI category properties	Musician category properties
dbpprop:style	dbpprop:genre
dbpprop:architecture	dbpedia-owl:genre
dbpprop:architectureStyle	dcterms:subject
dbpprop:architecturalStyle	
dbpprop:architectureType	
dbpedia-owl:architecturalStyle	

To obtain musicians we query DBpedia for the entities related to each of the categories stored in the music taxonomies through any property of a second predefined list (Table 3). Finally, similarly to POIs, we query DBpedia again to retrieve the musicians' metadata.

From the metadata gathered for the identified POIs and musicians, we select those that let build the semantic framework and networks explained in Section 3.1 and illustrated in Figure 2. Tables Table 4 and Table 5 show the considered properties for extracting place and date relations between POIs, cities and musicians from DBpedia.

Table 4. Place and date properties considered for retrieving POI metadata.

Building start date properties	Building end date properties	Opening date properties
dbpprop:year	dbpprop:built	dbpprop:foundationDate
dbpprop:years	dbpprop:completionDate	dbpprop:foundedDate
dbpprop:constructionDates	dbpprop:constructedDate	dbpedia-owl:foundingDate
dbpprop:constructionStartDate	dbpprop:created	dbpprop:inaugurationDate
dbpprop:startDate	dbpprop:creation	dbpprop:dateOpened
dbpprop:yearStarted	dbpprop:established	dbpprop:open
dbpedia-owl:buildingStartDate	dbpprop:establishment	dbpprop:opened
	dbpprop:yearBuilt	dbpprop:opening
	dbpprop:yearCompleted	dbpprop:openingDate
	dbpedia-owl:buildingEndDate	dbpprop:openToPublic
		dbpedia-owl:openingDate
		dbpedia-owl:openingYear

Table 5. Place and date properties considered for retrieving musician metadata.

Birth place properties	Birth date properties
dbpprop:placeOfBirth	dbpprop:born
dbpprop:birthplace	dbpprop:dateOfBirth
dbpedia-owl:birthplace	dbpprop:birthdate
Death place properties	dbpedia-owl:birthdate
dbpprop:placeOfDeath	dbpedia-owl:birthYear
dbpprop:deathPlace	**Death date properties**
dbpedia-owl:deathPlace	dbpprop:died
Origin place properties	dbpprop:dateOfDeath
dbpprop:origin	dbpprop:deathDate
dbpedia-owl:origin	dbpedia-owl:deathDate
Residence place properties	dbpedia-owl:deathYear
dbpprop:residence	**Activity date properties**
dbpedia-owl:residence	dbpprop:yearsActive
dbpprop:hometown	dbpedia-owl:activeYearsStartYear
dbpedia-owl:hometown	

3.3 Cross-domain semantic relatedness

Once we have extracted instances in a target domain that are somehow related to the given instance(s) in the source domain, we have to identify which of them have the highest semantic relatedness, in order to finally recommend them to the user. For this purpose, we have investigated several algorithms to measure the semantic relatedness between instances, and to generate a ranking of the most related instances with respect to the input instance(s) and to the user's preferences.

More specifically, we have evaluated two sets of semantic similarity metrics that suit two different types of information. On the one hand, we have tested *statistical similarities* that compute similarities between items from non-structured data, such as textual item descriptions. We call them as text-based semantic similarities, and present them in Section 3.3.1. On the other hand, we have tested *topological similarities* that compute similarities between items making use of structured metadata and semantic network topology. We call them as graph-based semantic similarities, and present them in Section 3.3.2.

3.3.1 Text-based semantic similarities

For this type of item semantic similarity metrics, we assume that items are described by means of text documents. To define them, we therefore use the terms "item" and "document" indistinctively.

Since text is an unstructured source of information, text-based semantic similarities between two documents (items) rely on some sort of statistical analysis, mainly term co-occurrence analysis. In particular, our approach utilizes Wikipedia as a source of universal and multi-domain knowledge in textual format. Since DBpedia is the Linked Data version of Wikipedia, we are able to match every item (semantic entity) extracted with our framework to its corresponding Wikipedia's text article.

As explained in Section 2.1.2, in order to compute text-based semantic relatedness, we first have to convert the documents into vector representations $\overrightarrow{d_n} = (d_{n1}, \dots, d_{nM})$, where d_{in} is a real number that measures the relative importance of term t_m to describe the content of document d_n, and M is the size of the documents' vocabulary, i.e., the number of distinct terms in the whole set of documents. More specifically, in the addressed case study, for each POI and musician we download its corresponding article in the English Wikipedia, and perform the following text preprocessing tasks:

1. *Document tokenization*. The whole text of a Wikipedia article is decomposed into individual terms, which are normalized to lower case.
2. *Stop-words filtering*. Words that are too common in English are filtered out, since they do not contain any valuable information for a specific document. Examples of such words are articles and prepositions such as *the, and, by*.
3. *Term stemming*. The lexical root of each word is extracted, reducing derived terms with different gender or number to the same form. This is necessary in order to reduce the size of the vocabulary, and increase the overlap between documents. Word stems are the smallest bits of information that are used to describe the documents, that is, the $\overrightarrow{d_n}$ vector representation.
4. *Term weight computation*. Each (stemmed) term is assigned a real number (weight) that measures its importance for describing a document in which it appears. For computing such weight, we have explored two strategies, namely the *binary weighting strategy* –which assigns a term with the values 1 or 0 if it appears or does not appear in a document, respectively–, and the *frequency-based weighting* –which assigns a term with its TF-IDF score.

To perform these text processing tasks, we used the `EnglishAnalyzer` from the Apache Lucene library[10], which is an information retrieval engine written in Java that provides tools for preprocessing documents in several languages, creating indices, and performing search queries.

With the built document vectors, we measure some of the vector similarities described in Section 2.1.2. In addition to such similarities, we implemented the well-known Dice, Jaccard and Overlap coefficients, which are also frequency-based similarities:

$$dice(d_i, d_j) = \frac{\overrightarrow{d_i} \cdot \overrightarrow{d_j}}{\|\overrightarrow{d_i}\| + \|\overrightarrow{d_j}\|}$$

$$jaccard(d_i, d_j) = \frac{\overrightarrow{d_i} \cdot \overrightarrow{d_j}}{\|\overrightarrow{d_i}\| + \|\overrightarrow{d_j}\| - \overrightarrow{d_i} \cdot \overrightarrow{d_j}}$$

$$overlap(d_i, d_j) = \frac{\overrightarrow{d_i} \cdot \overrightarrow{d_j}}{\min(\|\overrightarrow{d_i}\|, \|\overrightarrow{d_j}\|)}$$

We also implemented the LSA technique with different dimensions for the latent semantic subspace, in order to test the performance of a dimensionality reduction algorithm.

3.3.2 Graph-based semantic similarities

Graph-based semantic similarity metrics take into account the structure (topology) of the semantic network/graph of two concepts or nodes to compute the relatedness between them. In Section 2.1.3 we reviewed representative state of the art graph-based similarities, most of which assume that the exploited semantic network forms a taxonomy like WordNet. Our approach, in contrast, does not require a well-defined hierarchical structure, since the semantic DAGs it uses are built from root nodes –instances in the source domain– which are connected to other concepts using links that may be more generic than specialization (categorization) relations, and may belong to different domains. As a result, in such DAGs, notions like the least common subsumer for computing semantic relatedness cannot be applied directly.

Other similarity metrics like the Wikipedia link-based measure (WLM) (Milne & Witten, 2008) and the Linked-Data semantic distance (LDSD) (Passant, 2010) do not rely on a hierarchical structure. The WLM is not suitable for the semantic networks built in our framework, since we use a limited number of semantic relations and the number of common incoming/outgoing links between a POI and a musician is generally not large enough to compute reliable scores. The LDSD is, in contrast, a good candidate for our framework, as it is designed for Linked Data graphs, and does not require a class taxonomy in the underlying ontology. Its major drawback is that it does not use the whole graph structure, but only uses semantic paths up to distance two.

For our framework, we need a graph-based scoring method that: 1) measures the influence/importance of a set of nodes **relative to** another fixed "root" node, 2) is capable of handling **any type of link**, and not only taxonomical structures, and 3) uses the **whole graph structure** with all the types of graph paths, and without being limited to a particular path distance.

In this thesis we have focused on a **Spreading Activation algorithm** (Crestani, 1997), since it satisfies the three above conditions, and is a natural method for influence/relevance propagation on semantic networks. The algorithm propagates the initial score of a given source node through its (weighted) outgoing edges, updating the scores of its linked nodes. This is iteratively done for subsequent linked nodes until reaching certain target nodes, whose scores cannot be further

[10] Apache Lucene, http://lucene.apache.org

propagated because they do not have outgoing edges. More specifically, the weight of a node x in the DAG built from root r is computed as is computed as

$$\mathrm{rel}(x,r) = \sum_{y \to x} \mathrm{rel}(y,r) \cdot w(y \to x)$$

where $\{y \to x\}$ is the set of nodes linking to x, and $w(y \to x)$ is the weight of the edge that links y and x. The score of the root entity is initialized so that $\mathrm{rel}(r,r) = 1$. We note that since we are working with DAGs, we can find a topological sorting $(x_1, \ldots, x_L)$ of the nodes such that x_i *precedes* x_j in the ordering established by the directed edges when $i < j$. This ensures that if we compute the scores according to the ordering, the values $\mathrm{rel}(x_i, r)$ are already available when computing the score for node x_j, and thus a single iteration through the whole graph is sufficient to compute the semantic relatedness scores of all the nodes. Figure 8 shows the score propagation process according to the spreading activation algorithm for an example semantic network.

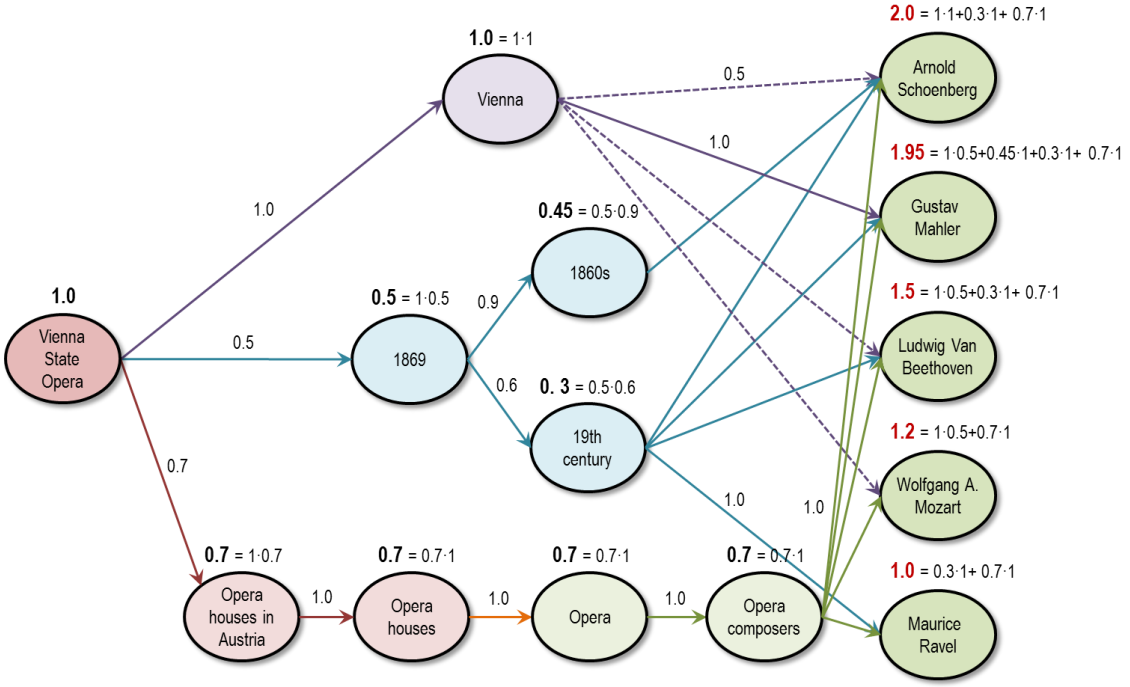

Figure 8. An example of score propagation in an instance sub-network of the Vienna State Opera; for the location paths, solid and dashed arrows represent `birth_place_of` and `death_place_of` properties, respectively.

Moreover, since the used graphs are built from a root node, and are restricted to related concepts, the above requirements are also implicitly fulfilled by metrics that compute global semantic relatedness scores on the sub-graph of scope. Formally, if we name $G(r)$ the semantic network built from root concept r, we define

$$\mathrm{rel}(v,r) = M_{G(r)}(v)$$

where $M_X(\cdot)$ is some graph global scoring/ranking algorithm applied over graph X.

As a baseline graph global ranking algorithm we have considered the **HITS algorithm** (Kleinberg, 1999), which exploits the whole structure of a graph assigning two scores to each node, namely *authority* and *hub* scores. The first score, called authority score, is based on the hub score of the nodes that link to it. Analogously, the second score, called the hub score, is based on the authority scores of the nodes linked from it. Thus, a node has a large authority score if it is linked

from many high scored hubs, and a node has a large hub score if it links to many high scored authorities:

$$A(i) = \sum_{j \to i} H(j) \qquad\qquad H(i) = \sum_{i \to j} A(j)$$

The final authority and hub scores are iteratively determined by computing the above formulas until convergence, which can be achieved by normalizing the scores after each iteration. In our approach, the ranking score assigned to an instance v is its final authority score on the subgraph built from root r, $G(r)$, since we are interested in identifying and retrieving the target domain instances with the highest structural influence within the cross-domain network:

$$\text{rel}(v, r) = A_{G(r)}(v)$$

The second baseline graph global ranking algorithm that we have used is the **PageRank algorithm** (Brin & Page, 1998). Similarly to HITS, PageRank exploits the link structure of a graph to determine the global importance of the nodes. PageRank computes only one score for each node by iteratively propagating the scores of the nodes linking to it. Following the random surfer model, the score of node is computed as follows:

$$\text{rel}(v, r) = P_{G(r)}(v) = (1 - d) \cdot \frac{1}{N} + d \cdot \sum_{u \to v} P_{G(r)}(u) \cdot \frac{w(u \to v)}{\sum_k w(u \to k)}$$

where d is a damping factor (set to 0.85 in our experiments) that reflects the probability of restarting the score propagation mechanism for the node v, and N is the total number of nodes in the graph.

Both HITS and PageRank are iterative algorithms that compute the relatedness scores upon convergence, usually until the relative improvement between two iterations is below some threshold. For DAGs, like the ones built in our framework, these algorithms behave similarly to the spreading activation algorithm with the notable difference that the scores are normalized and that there is a teleportation probability in the random surfer model, which means that every node can get a prior, non-zero weight. Therefore, we also have evaluated the undirected versions of these algorithms, adding backward edges letting the relatedness influence to *flow back* between nodes. More formally, whenever an edge $u \to v$ exists in the original graph, we also consider the opposite $v \to u$ with the same weight.

3.4 Personalized recommendations using cross-domain semantic relatedness

In the final stage of the proposed framework, we compute personalized recommendations of items in the target domain taking into account how semantically related they are to input information (preferences, context) of the user in the source domain. Specifically, we explore a ranking aggregation strategy in Section 3.4.1., and study personalized versions of the graph-ranking algorithms to compute user-specific semantic relatedness in Section 3.4.2.

3.4.1 Combining content-based recommendations with semantic similarities

A first approach we have explored to consider both item semantic relatedness and user preferences consists of generating content-based recommendations that only depend on the user profile, and combine the generated scores into a unique value that summarizes the overall utility. That is, we generate a ranking of items in the target domain according to the information of the user available also in the target domain, and then merge such ranking with the semantic relatedness scores (computed as explained in section 3.3) using a ranking aggregation method (Croft, 2002).

More formally, let $\vec{u} = (u_1, \ldots, u_k)$ be the vector representation of the user preferences on the target domain, described by k user features with associated weights u_j that summarize the past preferences of user u to items with attribute j. For instance, in our case study, the attributes that describe the musicians are their main music genres, and a user profile reflects if a user likes or not the considered music genres. An item profile $\vec{\imath} = (i_1, \ldots, i_k)$ is also a feature vector in which the weight of each component i_j represents to what degree attribute j describes item i. A content-based ranking of items for user u is generated by sorting the items according to some vector similarity metric $\mathrm{sim}(\vec{u}, \vec{\imath})$. Next, the semantically-enhanced recommendations are obtained by aggregating both the content-based ranking and the semantic relatedness ranking computed as described in section 3.3. In our implementation of the proposed framework, the score in the final ranking of item i for user u in context c is computed as a linear combination:

$$score(u, i, c) = \lambda \cdot \mathrm{rel}(i, c) + (1 - \lambda) \cdot \mathrm{sim}(\vec{u}, \vec{\imath})$$

where $\lambda \in [0,1]$ is a parameter that controls the importance of each source. Here and in the following the context refers to the root instance of the semantic networks, which can describe for example the location of the user (e.g. a POI) or related preferences from other domain. In the previous formula, note that if $\lambda = 0$ the contextual relatedness is ignored, and purely content-based item suggestions are generated. If $\lambda = 1$, the user preferences are not taken into account, and only the semantic relatedness plays a role in the item recommendations. If the item i does not appear in the semantic network for the context c, the relatedness value is assumed to be zero. As a practical note, both scores in the above formula are normalized to the $[0,1]$ interval. Hence, if $s(x, R)$ is the score of item x in ranking R, its normalized score is computed using the following formula:

$$\bar{s}(x, R) = \frac{s(x, R) - \min_y s(y, R)}{\max_y s(y, R) - \min_y s(y, R)}$$

In this way, score values across different rankings can be easily compared and effectively merged into a final ranking list.

3.4.2 Personalized graph-based semantic similarities

A second approach we have explored to consider both item semantic relatedness and user preferences for cross-domain recommendations focuses on computing a personalized version of the semantic relatedness graph-based metrics, by incorporating information about the user's preferences as prior knowledge into the semantic networks. That is, instead of computing $\mathrm{rel}(i, c)$, we compute personalized relatedness metrics $\mathrm{rel}(u, i, c)$.

As discussed in Section 3.1, prior relevance values can be assigned to both entity classes and instances. The pursued idea is to identify those nodes in the semantic networks for which user preference information is available, and increase their prior weight before running any ranking algorithm.

In the **Spreading Activation** algorithm, the personalized relevance value of an instance is propagated forward, and consequently the influence of related nodes is increased as follows:

$$\mathrm{rel}(u, i, c) = \lambda \cdot p(u, i) + \sum_{x \to i} \mathrm{rel}(u, x, c) \cdot w(x \to i)$$

where $p(u, i)$ is the prior weight that user u assigns to instance i, and $\lambda \in \mathbb{R}_+$ is a free parameter that controls the influence of the user preferences. In our case of study, we set $p(u, i) = u_i$ if node i represents a content item feature, i.e. a music genre, and $p(u, i) = 0$ otherwise. That is, we

only assign prior weights to instances that represent music genres, and set their value to the relative preference the user expressed for them.

We note that the spreading activation algorithm does not follow a probabilistic scheme, and $p(u,i) < 0$ can be set to penalize instances if there is evidence that the user dislikes the item. We also note that the relatedness score is unbounded, and that the above formula is not a convex combination of the prior and the propagated scores. Multiplying the second term by a factor $(1-\lambda)$ with $\lambda \in [0,1]$ would penalize scores propagated through longer paths, which resulted in lower performance on preliminary experiments. The results reported in chapter 4 were obtained using the unbounded version with $\lambda = 0.15$.

As a personalized version of **HITS**, we use an extension of the algorithm that includes prior probabilities proposed in (White & Smyth, 2003). The authority and hub scores in iteration $n+1$ are computed as:

$$A_{n+1}(i) = (1-d)\cdot p(u,i) + d\sum_{x\to i}\frac{H_n(x)}{\sum_y H_n(y)}$$

$$H_{n+1}(i) = (1-d)\cdot p(u,i) + d\sum_{i\to x}\frac{A_n(x)}{\sum_y A_n(y)}$$

The authors also give a probabilistic interpretation to the previous formulas, adapting the random surfer model so that $1-d$ is the probability of jumping to a random node according to the distribution $p(u,i)$. For the final relatedness value we again use the authority score.

The personalized version of **PageRank** is also obtained by incorporating prior vertex weights that represent the jumping probabilities in the random surfer model:

$$\mathrm{rel}(u,i,c) = P_{G(c)}(i) = (1-d)\cdot p(u,i) + d\sum_{x\to i} P_{G(c)}(x)\cdot\frac{w(x\to i)}{\sum_z w(x\to z)}$$

In both PageRank and HITS, the prior values $p(u,i)$ are now normalized so that $\sum_i p(u,i) = 1$.

When the source domain represents the user's context, this approach computes personalized recommendations only among items that belong to the semantic network associated to the source instance that describes the context. Thus, the approaches described in this section fall in the category of pre-filtering context-aware recommender systems, explained in Section 2.2.2.

Chapter 4. Experiments

In this chapter we present the experiments we conducted to evaluate the proposed framework in the POI-to-musician recommendation case study. These experiments consisted of several user studies in which participants were requested to evaluate the semantic relatedness metrics and cross-domain recommendation algorithms, developed for our framework.

We built three different Web applications to collect relevance assessments for the user studies, two for evaluating the semantic relatedness between POIs, musicians and music compositions, and another one to assess the personalized recommendations that exploit the semantic networks. The experiments were performed on a dataset with 25 POIs –of different types such as castles and palaces, religious buildings, and music venues– of 17 major European cities, and a total of 2080 musicians belonging 26 music genres.

In Sections 4.1 and 4.2 we report and analyze evaluation results obtained for the semantic relatedness and recommendation experiments, respectively. Next, in Section 4.3 we discuss the output of the user studies and relate it to the research goals we stated in Section 1.2.

4.1 Evaluating cross-domain semantic relatedness

In order to evaluate the quality of the built semantic networks, and the accuracy of the proposed **semantic relatedness metrics**, we conducted two user studies. The first study was designed to evaluate how people judge the semantic relations between POIs and retrieved musicians (Section 4.1.1), and the second study was aimed to evaluate if people consider compositions performed by the retrieved musicians as relevant for the POIs (Section 4.1.2).

Table 6 shows the list of cities and POIs that were selected for our evaluation dataset. A semantic instance network is built for each POI in order to find related musicians. On average, the instance networks of the 25 POIs consisted of 694.68 nodes (with 657.12 nodes representing musicians).

Table 6. Cities and POIs of the evaluation dataset.

City	POIs
Amsterdam	Canals of Amsterdam
Barcelona	Sagrada Familia, Casa Batlló
Berlin	Brandenburg Gate, Charlottenburg Palace
Brussels	Royal Palace of Brussels
Copenhagen	Christiansborg Palace
Dublin	Dublin Castle
Florence	Florence Cathedral
Hamburg	Hamburg Rathaus
London	Big Ben, Buckingham Palace
Madrid	Almudena Cathedral, Teatro Real, Las Ventas
Milan	Milan Cathedral, La Scala
Munich	Munich Frauenkirche
Paris	Eiffel Tower, Notre Dame de Paris
Prague	National Theatre (Prague)
Rome	St. Peter's Basilica, Colosseum
Seville	Seville Cathedral
Vienna	Vienna State Opera

4.1.1 Relatedness between POIs and musicians

Using the semantic relatedness algorithms presented in Section 3.3, we computed lists of ranked musicians for each input POI. Specifically, among the text-based approaches, we implemented the cosine, Dice, Jaccard, and Overlap vector similarities with/without LSA configurations. Among the graph-based approaches, we implemented both directed and undirected versions of the HITS and PageRank algorithms, as well as the proposed Spreading Activation algorithm. We also implemented a random ranking algorithm for comparison purposes. Hence, we implemented a total of 35 different ranking algorithms.

In order to limit the number of needed user assessments, we computed Spearman's rank correlation coefficient to find and filter out those algorithms that behave similar to others. The computed correlation values are reported in Appendix A. In general, we found low correlations between text- and graph-based algorithms, meaning that the two types of approaches rank musicians quite differently. For the text-based approaches, correlations between the implemented vector similarities were high (and moderately high when using LSA), especially between cosine, Dice, and Jaccard's metrics. Therefore, instead of evaluating all the algorithms, we selected representative algorithms for which the correlations were weak. In particular, we chose cosine and overlap, with their LSA variants using low ($k = 10$) and high ($k = 100$) k-dimensional latent spaces. For the graph-based approaches, both the directed and undirected versions of HITS and PageRank behaved in a moderately similar way, while the spreading algorithm had a quite different behavior. We considered the three graph-based algorithms for evaluation. Thus, we evaluated 12 different ranking algorithms. For each of these algorithms, and for each POI, we built an evaluation pool with the top 5 ranked musicians of the algorithm.

In the first user study, we asked participants to explicitly assess the relatedness between several POI-musician pairs. For such purpose, we used a Web application carefully designed, since assessing the quality of the relatedness (matching) between items from different domains is not an easy task. Hence, we developed a tailored interface (Figure 9) that may require considerable user effort, but let collect important and interesting information about the user-perceived quality of different musician matches for POIs.

Once logged in the Web application, during an evaluation session, a user was presented with a sequence of 10 POI-musician pairs, where the musicians were obtained by one of the 12 evaluated ranking algorithms. The user was not aware of what algorithm was used to identify the musician recommended for a POI.

The information describing each POI-musician pair was presented in a structured way, according to the representation in the class network –location, date, and category relations were clearly separated. A user was asked to carefully check the presented information, and assess whether the musician was actually related to the POI, and if it was, to specify which parts of the structured musician description were contributing, and in which degree, to the match (right part of the Web application's interface, Figure 9). We aimed to understand which types of semantic paths, linking POIs to musicians (i.e., either location, date, or category paths) contributed more to the matches, and were better appreciated by the users.

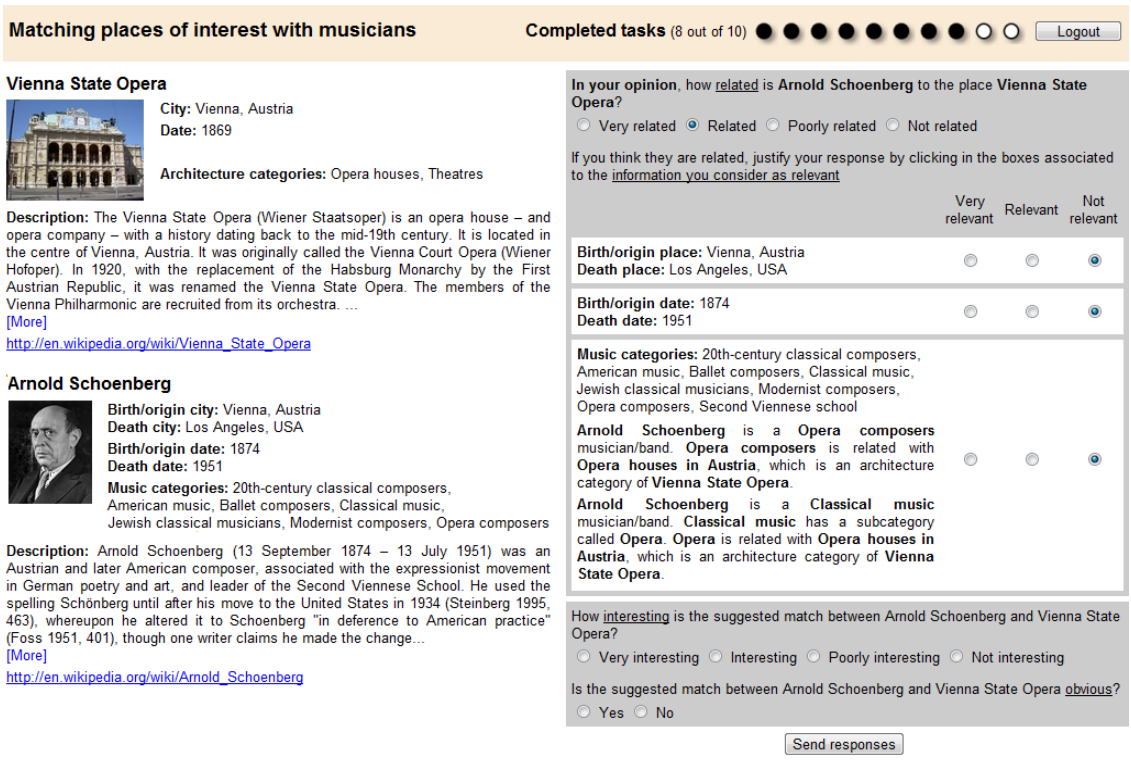

Figure 9. Screenshot of the Web application used to assess the knowledge-based relatedness of POIs and musicians.

A total of 97 users participated in the study. They were PhD students and academic staff recruited via email, and covered an ample spectrum of ages and nationalities. The users provided 3666 assessments for 1222 distinct POI-musician pairs (note that a musician may match the same POI using various ranking algorithms). Each of the 1222 distinct pairs was assessed by at least 3 users. The Fleiss' Kappa correlation coefficient of the relatedness assessments per POI was 0.675, meaning a substantial agreement among users.

Table 7 shows the average precision and nDCG values for the top 5 ranked musicians obtained by each algorithm. Precision at rank level k (P@k) measures the proportion of relevant items on the top k retrieved musicians, whereas Mean Average Precision (MAP) reports the precision value after each relevant musician is retrieved, averaged over all POIs. The normalized Discounted Cumulative Gain (nDCG) metric takes into account graded relevance (as opposed to binary relevant/not relevant), and is sensitive to the order in which musicians are ranked.

From the table, we can see that the spreading algorithm consistently outperformed the rest of the algorithms for all metrics, followed up by the text-based cosine and overlap metrics, and the undirected version of the PageRank algorithm. As expected, all of the algorithms achieved much higher performance than the random algorithm.

Table 7. Performance of the 12 selected algorithms for semantic relatedness. The spreading algorithm performed statistically significantly better (Wilcoxon signed rank test, $p<0.01$) than the rest of the methods, except in those cases marked with an asterisk.

Ranking algorithm	MAP	P@1	P@2	P@3	P@4	P@5	nDCG
Random	0.0159	0.0267	0.0267	0.0444	0.0500	0.0480	0.2535
Cosine	0.3153*	0.5600*	0.6600*	0.6400*	0.6700*	0.6400*	0.6338*
Cosine with LSA (k = 10)	0.2590	0.6800*	0.6400*	0.6400*	0.6400*	0.6000*	0.5544*
Cosine with LSA (k = 100)	0.2426	0.4800	0.5000	0.4667*	0.5000*	0.4800	0.5408*
Overlap	0.2982*	0.6400*	0.6000*	0.5467	0.5600*	0.5760*	0.6160*
Overlap with LSA (k = 10)	0.1172	0.4400	0.3800	0.3600	0.3200	0.3120	0.4045
Overlap with LSA (k = 100)	0.1671	0.4400	0.4000	0.3867	0.3700	0.3600	0.4547
HITS	0.1936	0.4000	0.3800	0.3467	0.3300	0.2960	0.4761
HITS (undirected graph)	0.1815	0.2400	0.2400	0.3067	0.3600	0.3200	0.4544
PageRank	0.1936	0.4000	0.3800	0.3467	0.3300	0.2960	0.4761
PageRank (undirected graph)	0.2846	0.6800*	0.5400	0.4933	0.4800	0.4480	0.5711
Spreading	**0.4138**	**0.9200**	**0.8600**	**0.8000**	**0.7900**	**0.7360**	**0.6670**

To show that the users' appreciations of semantic relatedness between POIs and musicians were not biased to certain types of places of interest such as music venues, Table 8 shows the average performance values obtained by the spreading algorithm for various main POI types, namely *castles and palaces*, *music venues* (opera houses, theaters), *religious buildings* (cathedrals, churches), and *other POIs* (e.g. gates, towers, monuments). We can see that there is no particular POI type in which the spreading algorithm performs consistently better, and we conclude that users assessed the relevance of the proposed musicians independently of the type of the POI.

Table 8. Performance of the spreading algorithm for different types of POIs.

POI type	MAP	P@1	P@2	P@3	P@4	P@5	nDCG
Castles and palaces	0.3632	0.8333	0.7500	0.7778	0.7500	**0.8000**	0.5930
Music venues	**0.4639**	0.7500	0.7500	0.7500	0.7500	0.7500	**0.7661**
Religious buildings	0.4568	**1.0000**	**1.0000**	**0.8334**	**0.8438**	0.7250	0.6606
Other POIs	0.3794	**1.0000**	0.8571	0.8095	0.7857	0.6857	0.6811

We compared the performance of all the algorithms for the different types of POI. The results can be found in Appendix B. From them, we see that the spreading algorithm performs best for all POI types except for the *music venues*, in which the text-based semantic relatedness methods using cosine and overlap metrics obtained the higher precision values. This is presumably due to the similar vocabulary used to describe both types of items. We recall that the text-based approaches use the Wikipedia's articles of POIs and musicians to compute similarities between them.

In order to understand if the proposed knowledge representation in the framework is indeed effective, we analyzed the characteristics of the relevant musicians in their respective semantic networks, measuring in which degree the different types of paths contributed to the POI-musician matches. We can see in Figure 10 that more relevant musicians have on average a high number of paths connecting them to the root POI. The average length of these paths is shorter as well. Another difference worth noticing is that the relevant musicians have on average one less category path, and that their length is shorter. Intuitively, we found that connections through deeper categories in the architecture and music taxonomies lead to longer, less informative paths that do

not contribute to the overall relevance. It is also evident from the figure that having at least one location path seems a necessary condition for relevant musicians.

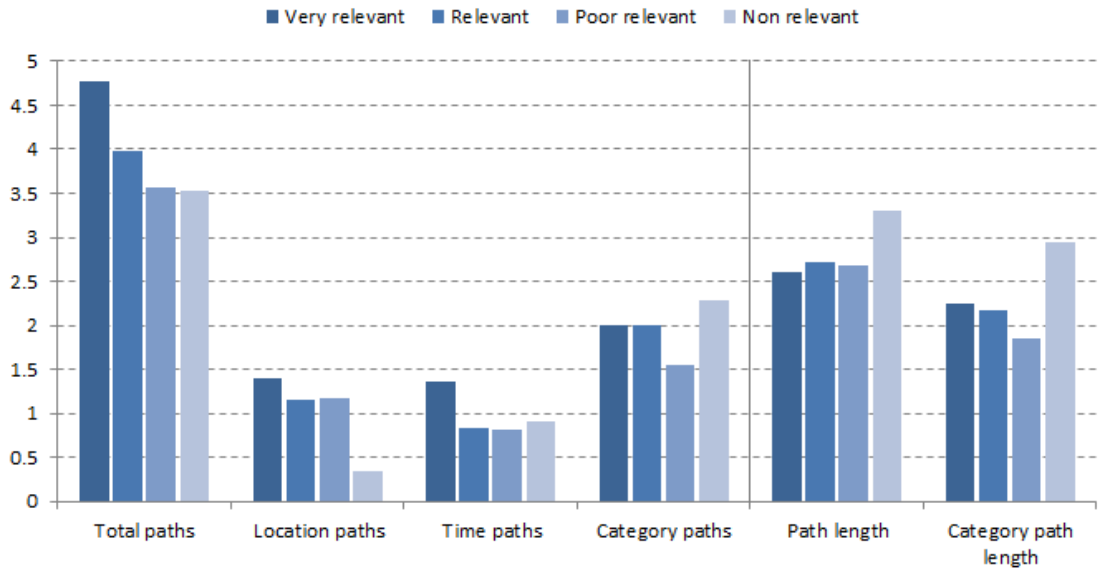

Figure 10. Average number and length of paths from the evaluated POIs to their relevant/non-relevant musicians.

Finally, Table 9 shows the percentages of the related musicians, according to the spreading algorithm, which were assessed as (non) interesting and (non) obvious. It can be seen that a high percentage (73.4%) of the relevant matches retrieved by the proposed approach were indeed interesting for the users. Moreover, a high percentage of them were evaluated as non-obvious (88.6%). These, together with the previous results, show that musicians suggested by our approach were semantically related and relevant for the input POIs.

Table 9. The users' interest for relevant musicians.

	Non interesting	Poor interesting	Interesting	Very interesting	Non obvious	Obvious
Poor relevant	30.71%	**57.48%**	11.81%	0.00%	44.88%	**55.12%**
Relevant	0.00%	20.25%	**73.42%**	6.33%	**88.61%**	11.39%
Very relevant	0.00%	0.00%	35.63%	**64.37%**	**79.31%**	20.69%

4.1.2 Relatedness between POIs and music compositions

In order to further analyze the semantic relatedness of the POI-musician matches obtained by our graph-based approach, we conducted a second user study in which participants were requested to assess the suitability of music compositions of the retrieved musicians for the corresponding POIs.

For such purpose, we downloaded a music track for each musician by taking the top-ranked result returned by the YouTube API. This ensured the collected music tracks to be representative of the musicians in our dataset. In this case, we considered 4 competing algorithms to find a musician (and its music composition) matching any given POI.

We designed a simple Web application (Figure 11) aimed to assess the quality of the matching between a POI and the music of a musician suited to a POI. We wanted to understand if the music

of the matching musician was evaluated by the users as suited for the POI, whatever is the idea that a user may have about what music suited for a place is. For instance, if Schoenberg is found by our algorithms as a good match for Opera House in Vienna, is Pierrot Lunaire, a popular composition by this author, a suited music?

During a single evaluation case a user was presented with a POI and a list with one composition of each musician selected by any of the four evaluated ranking algorithms. The order of the compositions was randomized, and the user was not aware of the ranking algorithms that were used. The user was then asked to read the description of the POI, listen to the compositions, and select those compositions that in her opinion suited the POI.

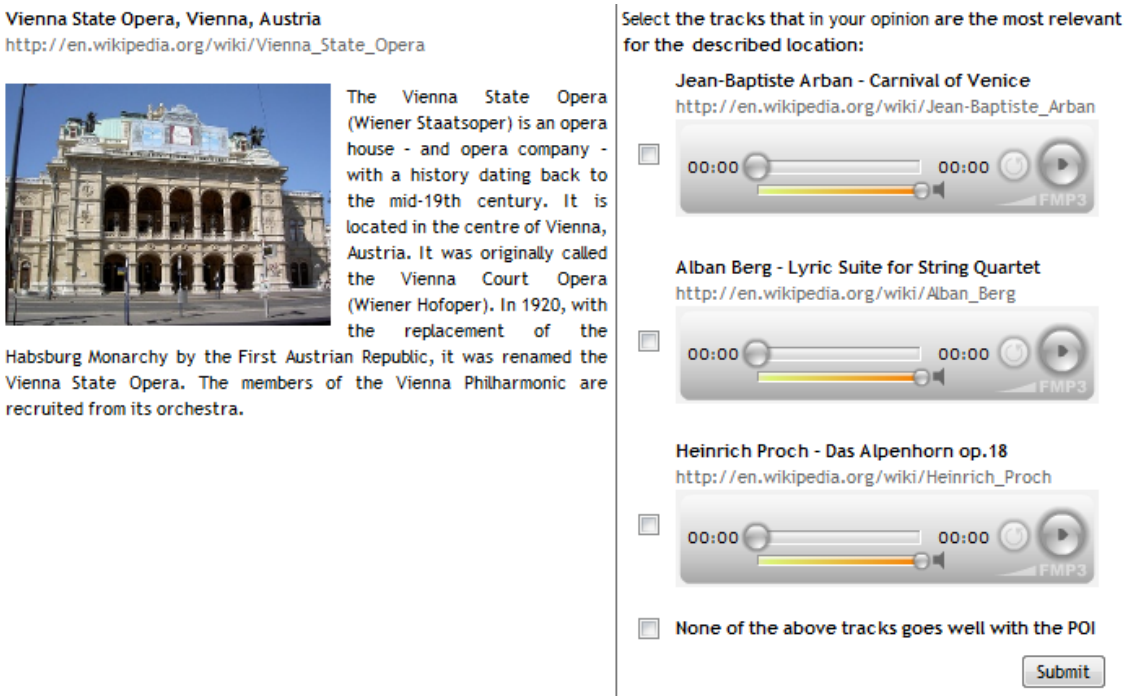

Figure 11. Screenshot of the Web application used in the matching music to POIs evaluation study.

Moreover, prior to performing the evaluation, we asked the users to enter their music genre preferences (Figure 12). This was done to measure the influence of the users' music preferences on their decisions. The genre taxonomy was selected based on the musicians in our dataset, and included Classical, Pop, Medieval, Opera, Rock, Ambient, Folk, Hip Hop, Metal, and Electronic music genres.

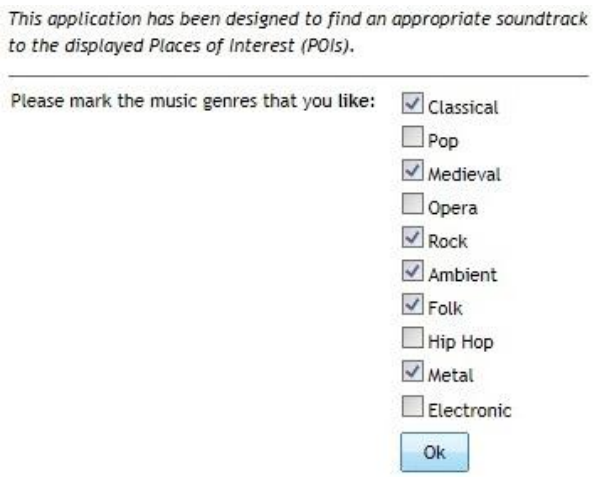

Figure 12. Screenshot of the interface for collecting the users' music genre preferences.

A total of 61 users participated in the second study. As in the first case, they were PhD students and academic staff recruited via email (some of them also participated in the previous study). 1125 evaluation sessions were performed (i.e., a POI shown to a user), and 1258 tracks were selected by the users as well-suited for POIs. Figure 13 shows the performance of the approaches, computed as the ratio of the number of times a track produced by each method was considered as well-suited over the total number of evaluation sessions (i.e., the number of the times that a track was also offered to the user as a potential match). All approaches performed statistically significantly better than the random track selection ($p<0.01$ in a two-proportion z-test). Moreover, again, the spreading activation method outperformed the others ($p<0.01$).

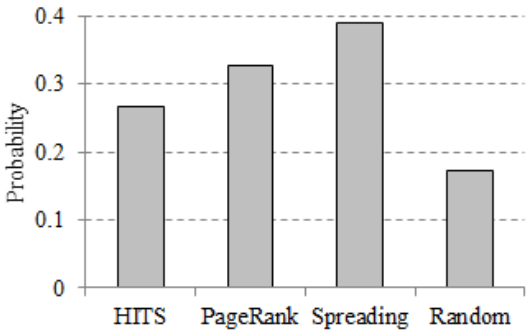

Figure 13. Selection probability of the music recommendation approaches.

While in the described study participants received music recommendations that matched the displayed POI, the recommendations were not adjusted to each user's music genre preferences. Therefore, we expected to observe a variance in the results depending on the users' music tastes. To confirm this, we analyzed the influence of the users' genre preferences on the music tracks that they selected as well-suited for the POIs.

In the following, $TrkOK$ represents the condition that a track was marked by a user as well-suited for a POI, $GTrk = g$ is the condition that the genre of a track is g, $g \in uPref$ represents the condition that a user has included the genre g in her genre preferences, and $g \notin uPref$ the condition that a user has not included the genre in her preferences.

As a baseline for the analysis, for each genre, we computed the probability that a music track is of genre g, given that it was marked as suited for a POI, $P(Gtrk = g|TrkOK)$, as the ratio of the number of tracks of genre g selected as well-suited for the POIs over the total number of tracks selected as suited for POIs (1258). Then, to check the deviation from this baseline produced by the users' preferences, we computed $P(GTrk = g|TrkOK, g \in uPref)$ and $P(GTrk = g \,|\, TrkOK, g \notin uPref)$.

We found that for the Classical, Medieval and Opera music genres, the deviation from the baseline is significant ($p<0.01$ in a two-proportion z-test). For Classical music, the probability that a user who likes this genre will mark a classical track as well-suited for a POI was higher, and in the opposite case – lower. For other music genres the deviation from the baseline probabilities was not significant. This indicates that the decision to mark a music track as suited for a POI is influenced by the user musical tastes. The fact that this depends on the music genre could be related to the knowledge of that genre of music. In fact, Medieval and Opera music was rarely among the preferred music of our sample.

We also computed the conditional probability for a track to be selected as suited for a POI, given that its genre is g, $P(TrkOK|GTrk = g)$, as the ratio of the number of tracks of genre g selected as suited over the total number of times a track of genre g was displayed during the evaluation. The obtained results show that Medieval and Opera tracks were most often selected by the users (*0.49* probability), followed by Ambient (*0.36*) and Classical (*0.35*) music. Hence, this type of music was not liked by the users (in general) but was often considered as suited for the POI.

Finally, to check the deviation from this baseline produced by the users' preferences, we computed the probabilities $P(TrkOK \mid GTrk = g, g \in uPref)$ and $P(TrkOK \mid GTrk = g, g \notin uPref)$. From these probabilities the effect of user preferences was again evident for Classical and Medieval music – the probability that a user will select a Classical/Medieval track as well-suited for a POI was significantly ($p<0.05$) higher if the user liked these genres, compared to a user who did not.

We can thus confirm that for certain genres, in addition to the semantic matching between POIs and musicians, there is a clear effect of user preferences on the decision for considering music to go well with a POI: if a user likes a music that is generally suited for the POI (e.g. classical), the probability that she will evaluate that music as suited will increase as well. But, if a music genre is not generally suited for the POIs (e.g. rock), then even if the user likes that music, it does not suffice to make this music type suited for a POI.

As the presented analysis shows it is important to consider user preferences and therefore, in the following experiment we investigate how to take them into account when suggesting music suited for a POI.

4.2 Evaluating recommendations using cross-domain semantic relatedness

After evaluating the proposed semantic relatedness metrics, we conducted a user study to analyze which of the recommendation algorithms presented in Section 3.4 more effectively exploits the user's preferences together with the above metrics. The purpose of the study thus was twofold: first, to check if users found musicians semantically related to the POI valuable or if they preferred to rely only on their preferences, and second, to empirically compare the personalized versions of the proposed recommendation algorithms. More specifically, we evaluated the linear combination approach (Section 3.4.1) with different values of its combination parameter λ, and the personalized version of the spreading algorithm (Section 3.4.2), since its non-personalized version performed better than the other semantic relatedness metrics evaluated.

In the study we considered the 26 music genres used in the Last.fm system. Each of these genres was composed of a set of popular tags, publicly available in that system. We built genre-based profiles for the musicians of our database as follows. We extracted the musicians' tag-based profiles from Last.fm, and transformed them into genre-based profiles by computing cosine similarities between the tag-based profiles of each genre and musician. More specifically, the weight of genre g for a musician m was set as $cos(\vec{g}, \vec{m})$, where $\vec{g}$ and $\vec{m}$ are the tag-based profiles of g and m respectively. Initially, we also assigned weights to the tags of each genre and musician by applying TF-IDF mechanisms with the collections of tags retrieved for the genres and musicians separately.

Afterwards, for the experiment, we built a Web application (Figure 14) in which users were first asked to enter their preferences on the set of considered music genres. For the linear combination recommendation approach, these preferences constituted the user profiles used by the content-based component, which directly computed content-based similarities between users and musicians. For the personalized spreading algorithm, in contrast, the preferences had to be automatically mapped to DBpedia entities representing music genres. For instance, the *cat:Punk_rock* entity in DBpedia was mapped with the Last.fm genres *punk* and *rock*.

In particular, the 26 Last.fm genres were mapped with a total of 217 DBpedia entities of our database like follows. Let $G = \{\overrightarrow{g_1}, \dots, \overrightarrow{g_K}\}$ be the set of music genres from the Last.fm, for which the users expressed their preferences, and $C = \{\overrightarrow{c_1}, \dots, \overrightarrow{c_L}\}$ the set of DBpedia concepts that represent the 217 music genres we want to map to. Each of the $\overrightarrow{g_k}$ is now a musician-based vector instead of a tag-based profile as previously, that is, its j-th component g_{kj} is the similarity between genre g_k and musician m_j, computed as described before. Likewise, each $\overrightarrow{c_l}$ is also a musician-based profile, in which the j-th component c_{lj} is 1 if the musician m_j is related to concept c_l in DBpedia and 0 otherwise. The similarity between DBpedia genre $\overrightarrow{c_l}$ and Last.fm genre $\overrightarrow{g_k}$ is computed as

$$\text{sim}(c_l, g_k) = \overrightarrow{c_l} \cdot \overrightarrow{g_k} = \sum_{j=1}^{M} c_{lj} \cdot g_{kj}$$

where M is the number of musicians. The assumption is that the mapping is stronger if many musicians are related to both genres. Let S be the genre similarity matrix whose (l,k)-th entry is defined as $s_{lk} = \text{sim}(c_l, g_k)$. Consider now the user profile $\overrightarrow{u^G} = (u_1^G, \dots, u_K^G)$, where u_k^G is the preference of user u towards Last.fm genre g_k. In our experiment, $u_k^G \in \{0,1,2,3\}$, so that the users can express different degrees of preference (see Figure 14). The final profile based on DBpedia genre concepts is computed as $\overrightarrow{u^C} = S \cdot \overrightarrow{u^G}$.

Then, for each user, her preferred DBpedia genres were assigned a weight in the instance networks of the evaluated POIs, which was set from the corresponding component in the user profile. Let c_l be a DBpedia concept that represents a music genre in the semantic network, then the prior weight for the personalized graph-based semantic relatedness algorithms (Section 3.4.2) is taken as $p(u, c_l) = u_l^C$.

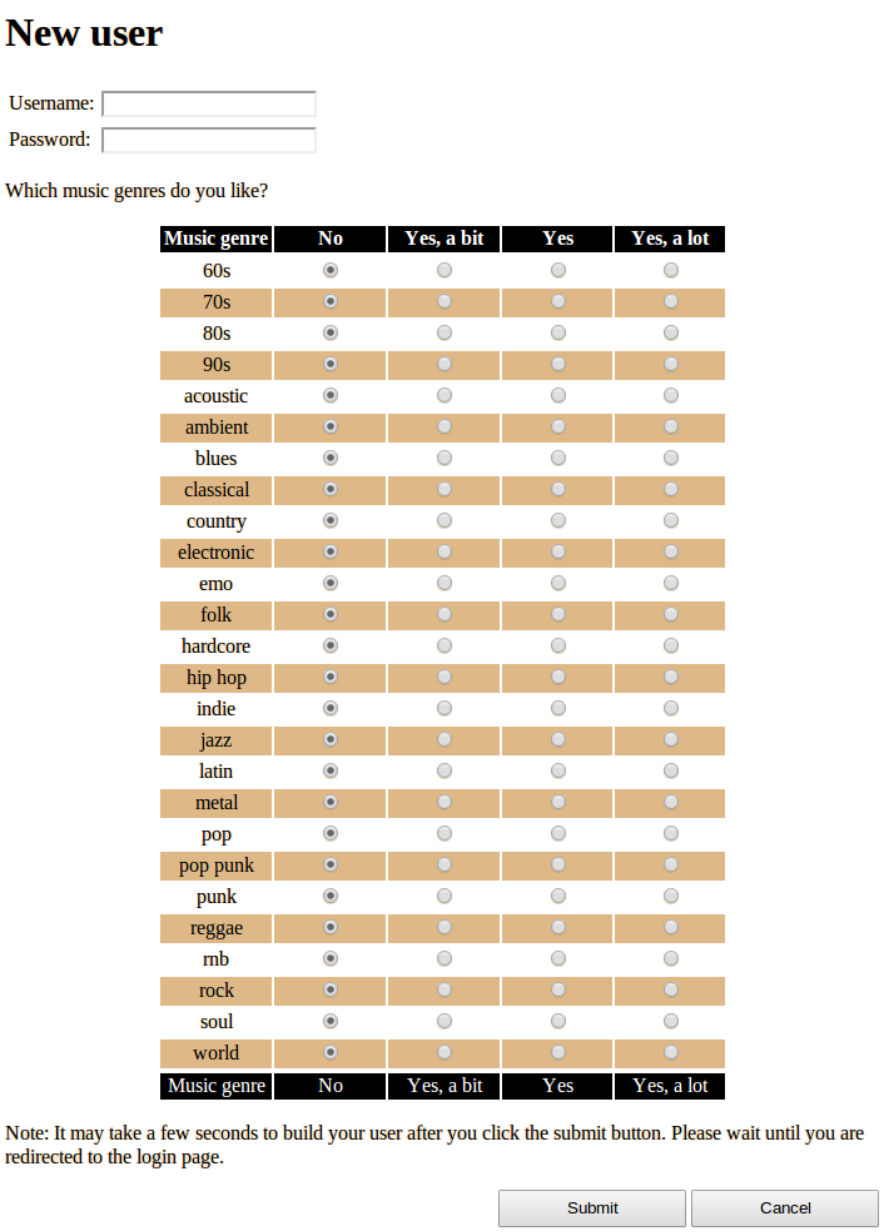

Figure 14. Screenshot of the Web application screen in which users entered their music genre preferences.

After entering her music genre preferences, a user was requested to assess pools with the top 5 ranked musicians retrieved by the evaluated algorithms for each POI. More specifically, for each pair $(user, POI)$ we selected the top 5 musicians retrieved by the linear combination algorithm with $\lambda \in \{0.25, 0.5, 0.75, 1\}$, and by the personalized spreading algorithm with $\alpha = 0.15$, as usually found in the literature. We note that the case $\lambda = 1$ corresponds to a purely content-based recommendation approach that ignores the semantic relatedness between POIs and musicians. Each pool of musicians was presented together with basic information about the POI. To better motivate our case study, users were asked to imagine they were visiting a particular POI wearing headphones, and were requested to choose which of the presented musicians they would listen to in such situation. Since the personalization process was performed at music genre level instead of at musician level, the users were also asked to indicate if they were already familiar with each particular musician. A screenshot of the evaluation application is shown in Figure 15.

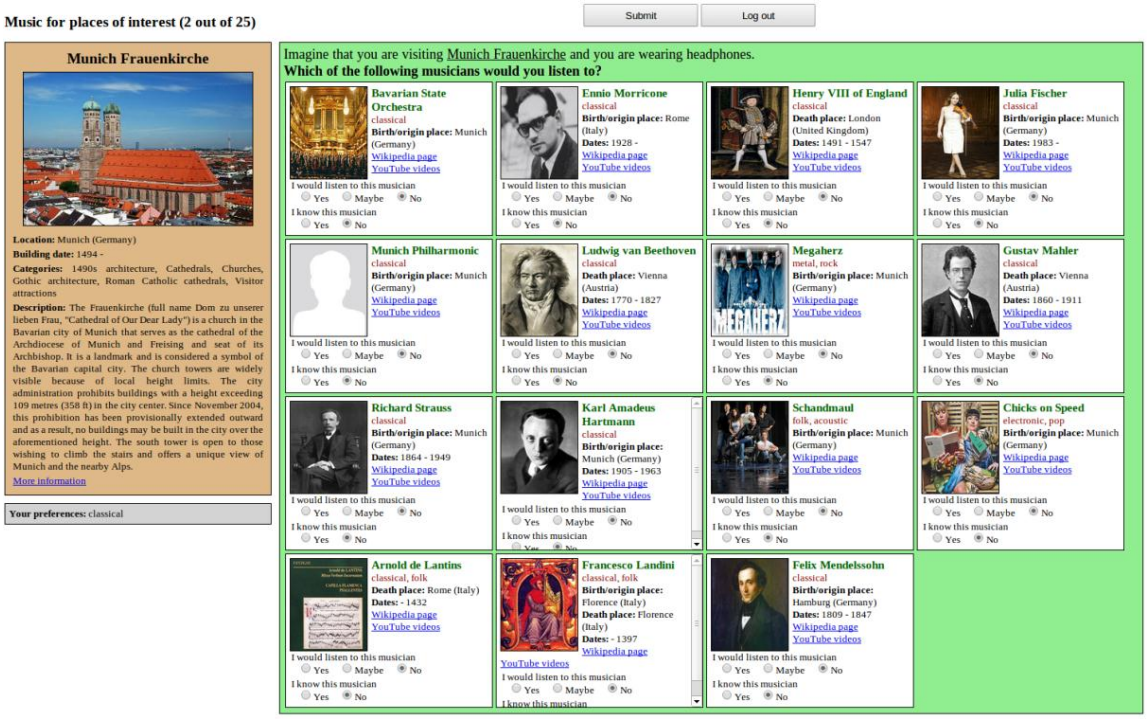

Figure 15. Screenshot of the personalized Web application to evaluate musicians matched with Frauenkirche Munich Cathedral.

A total of 17 users participated in this study, completing 190 evaluation sessions, and providing 1482 POI-musician relevance assessments. Some of the users found the evaluation somehow difficult because they were not familiar with many of the presented musicians. Although links to the Wikipedia's articles and YouTube videos of the musicians were provided, the task was difficult for the users, who had to identify non-trivial semantic relations between POIs and musicians.

Table 10 shows the precision and nDCG values achieved by the algorithms averaged over all users and POIs. We can see that in general the personalized spreading algorithm performed the best, followed closely by the linear combination with $\lambda = 0.25$, which assigns most of the weight to the semantic relatedness score. Furthermore, its performance decreased as the value of λ increased, which is an indication that the users indeed found the semantically related musicians valuable, regardless of their music tastes.

To check if the low precision achieved by the content-based algorithm was due to the unfamiliarity of the users with the musicians, we also computed the precision values for both known and unknown musicians. Again, the content-based algorithm still performed the worst, but in cases where the users known the musicians, they gave more importance to their preferences, and the linear combination algorithm with $\lambda = 0.25$ and $\lambda = 0.5$ outperformed the personalized spreading algorithm. This can be explained by the fact that the personalized version of the spreading algorithm is a pre-filtering context-aware recommendation algorithm, which is only able to suggest musicians that are related to the POI that may not overlap with the users' favorite musicians.

Table 10. Average performance of the personalized recommendation algorithms. When $\boldsymbol{\lambda = 1}$ the semantic relatedness is ignored, and the recommendations are purely content-based. All the methods performed statistically significantly better than the purely content-based approach (Wilcoxon signed rank test, p<0.01) in each set of considered musicians, except those marked with an asterisk.

Considered musicians	Algorithm	MAP	P@1	P@2	P@3	P@4	P@5	nDCG
All musicians	$\lambda = 1$	0.0261	0.0167	0.0278	0.0259	0.0278	0.0289	0.0371
	$\lambda = 0.75$	0.0888	0.1111	0.0778	0.0630	0.0611	0.0567	0.1104
	$\lambda = 0.5$	0.1944	0.1833	0.1667	0.1481	0.1500	0.1422	0.2498
	$\lambda = 0.25$	0.2589	0.2389	0.1889	0.1796	0.1958	0.1978	0.3241
	Spreading	**0.2856**	**0.2556**	**0.2222**	**0.2130**	**0.2083**	**0.2067**	**0.3440**
Known musicians	$\lambda = 1$	0.0263	0.0085	0.0212	0.0198	0.0212	0.0254	0.0399
	$\lambda = 0.75$	0.1132	0.1356	0.0932	0.0791	0.0678	0.0576	0.1398
	$\lambda = 0.5$	0.2230	**0.1949**	**0.1525**	**0.1271**	**0.1314**	0.1237	0.2743
	$\lambda = 0.25$	**0.2232**	0.1780	0.1314	0.1102	0.1208	**0.1271**	**0.2812**
	Spreading	0.1884	0.1356	0.0975	0.0904	0.0911	0.1153	0.2487
Unknown musicians	$\lambda = 1$	0.0114	0.0111	0.0139	0.0130	0.0139	0.0122	0.0152
	$\lambda = 0.75$	0.0195*	0.0222*	0.0167*	0.0111*	0.0167*	0.0189*	0.0270*
	$\lambda = 0.5$	0.0678	0.0556*	0.0667	0.0648	0.0639	0.0611	0.0958
	$\lambda = 0.25$	0.1385	0.1222	0.1028	0.1074	0.1167	0.1144	0.1716
	Spreading	**0.1883**	**0.1667**	**0.1583**	**0.1537**	**0.1486**	**0.1311**	**0.2175**

To further understand the behavior of the personalized spreading algorithm, we analyzed the popularity of each of the Last.fm genres in the semantic networks of the 25 POIs that we considered in the study. We computed the popularity of a genre within a semantic network by aggregating its similarity values towards the DBpedia entities in the graph it is mapped to. Specifically, the popularity of Last.fm genre g_k within network N is computed as

$$\text{popularity}_{\text{networks}}(g_k, N) = \sum_{c \in V(N) \cap C} \text{sim}(c, g_k)$$

where $V(N)$ is the set of nodes in the semantic network N, C is the set of DBpedia concepts that represent music genres, and $\text{sim}(c, g_k)$ is the similarity between Last.fm genre g and DBpedia genre c, computed as before. The overall popularity is the sum over the semantic networks of the 25 considered POIs.

We also computed the popularity of a genre within user profiles as the aggregated relevance all users gave to it:

$$\text{popularity}_{\text{users}}(g_k) = \sum_{u \in \mathcal{U}} u_k^G$$

where we used the same notation as earlier for u_k^G and $\mathcal{U}$ is the set of all users in our experiment.

Figure 16 shows the relative popularity of each genre, among both the semantic networks and the user profiles. We can see that the distribution is not uniform, and that there is a bias in the semantic networks towards some genres, specially *classical* and *rock*. Also, the genres most selected by the users have a considerable popularity among the semantic networks. This means that the preferences of most of the users actually modify prior weights in the semantic networks, and therefore the spreading algorithm is able to compute personalized relatedness scores. On the other hand, genres sometimes selected by the users like *70s* and *world* are not popular in the networks, and thus the scores of the personalized and non-personalized versions of the spreading algorithm will be similar.

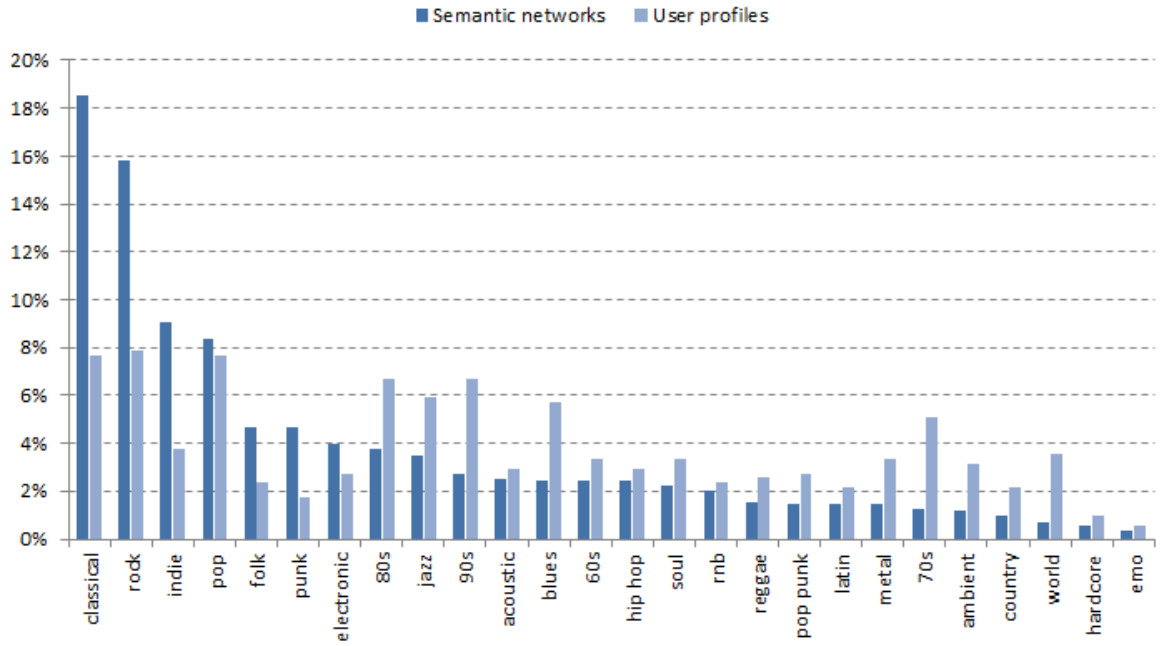

Figure 16. Popularity of the music genres in the semantic networks and user profiles.

To analyze the impact of the difference in popularity between the users' preferred music genres and the semantic networks, Figure 17 shows a comparison between the personalized and non-personalized scores computed by the spreading algorithm for the users who participated in the evaluation. In the figure, a data point corresponds to the scores of a musician for a POI given by the personalized and non-personalized spreading algorithm, and the colors represent the different users. Points in the diagonal correspond to cases where the algorithm was not able to perform any personalization, and since we are not penalizing any genre, no data points are at the left side of the diagonal. That is, a personalized score is always greater than or equal to its corresponding non-personalized score. We can see that for several users the personalized score is significantly increased in some POIs, while the effect is moderated in most cases –those that are close to the diagonal. More specifically, the average difference between the personalized and non-personalized scores is 0.341, and only 39.33% of the musician recommendations were affected by the personalization component. This may be due to the bias in popularity existing in the selected POIs for the evaluation towards certain genres, as shown in Figure 16.

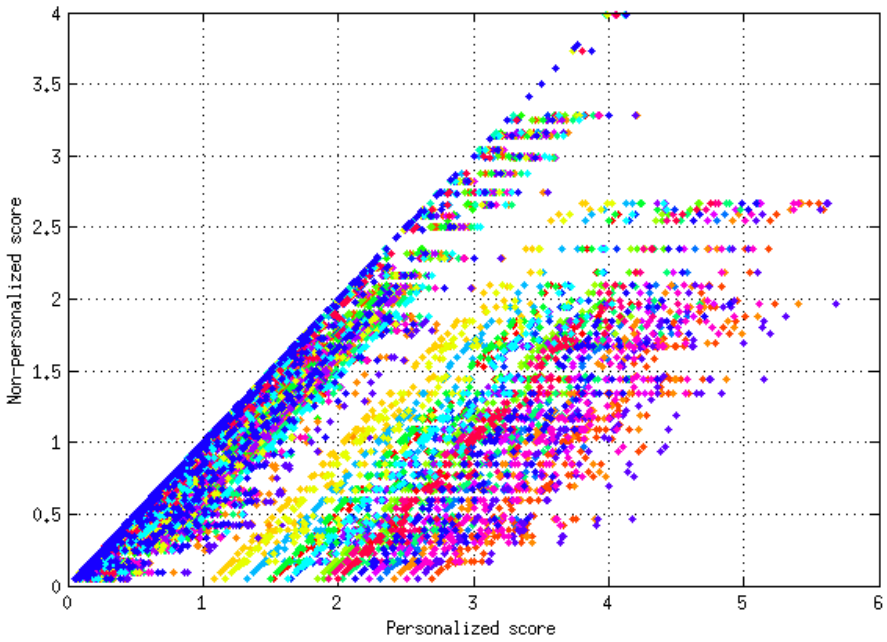

Figure 17. Comparison of the personalized and non-personalized scores from the spreading algorithm.

4.3 Discussion

As presented in this chapter, to validate our semantic-based framework for cross-domain recommendation, we conducted three user studies in the POI-to-musician recommendation case study. In the first experiment, addressing the research question RQ3, we evaluated the **quality of semantic relatedness metrics**, proving that the metric given by the proposed spreading algorithm was able to achieve high accuracy –between 70% and 90%– on retrieving musicians semantically related to the 25 POIs considered. Moreover, we proved that musicians which users stated as most relevant for the input POIs were connected through more, meaningful paths to such POIs in the corresponding semantic networks. Addressing the research questions RQ1 and RQ2, we thus confirmed that semantic networks built upon the DBpedia ontology can be used to identify, represent, and relate concepts from different domains in an effective way.

Specific for the addressed case study, in the second experiment, we evaluated the **relevance of music composed by the musicians matching a POI**, obtaining results in accordance with the first experiment, and confirming that popular tracks of the suggested musicians were also found by the users as relevant for the input POIs. Furthermore, we also analyzed the effect of the users' music preferences on their musician relevance judgments, proving that even if the users liked certain types of music, they did not find such music as relevant for a POI if it was not semantically related enough to the POI.

Finally, in the third experiment, we evaluated the developed algorithms for making **recommendations integrating user preferences with semantic relatedness information**. In the addressed case study, we proved that users in general gave more importance to recommended musicians semantically related to a POI over those recommended based on the users' music preferences, further confirming the results obtained in the second experiment, and directly addressing the research question RQ4. In particular, a personalized version of the proposed spreading algorithm outperformed a purely content-based algorithm, achieving over 7 times more precision than the latter. Nonetheless, we believe that there may be a bias to certain types of music for the selected POIs, and this may affect the coverage and limit the effectiveness of the personalization strategies.

Chapter 5. Conclusions and future work

5.1 Conclusions

Research in the area of RSs has traditionally focused on the task of recommending items from a single, limited domain. Recently, some work has been done addressing the problem of suggesting items in a target domain by exploiting user preferences, item attributes, or domain knowledge in different, but somehow related source domains. Most of the initial efforts on the cross-domain recommendation setting have applied CF techniques to overcome the existing item heterogeneity across domains, but require an enough overlap of user preferences on the considered domains. In a realistic situation, such overlap is unlikely, but we believe it could be mitigated by means of knowledge-based RSs, which make use of explicit information about the domains and their relations.

In this thesis, we have developed a framework that exploits information acquired from knowledge bases available in Linked Data to establish semantic networks among concepts of related domains, and to exploit such networks for personalized and context-aware cross-domain recommendation purposes. The work done has let us to achieve the following research contributions:

- Regarding the **representation of items and relations from different arbitrary domains** (RQ1), we have proposed an ontology-based knowledge representation composed of semantic networks at two levels. First, a class-network level in which principal class concepts from the considered domains are linked by means of a limited number of semantic properties. Second, an instance-network level in which classes are automatically instantiated as items semantically related to a given item in a source domain.
- Concerning the **extraction of knowledge to identify items in a target domain related to items in a source domain** (RQ2), we have developed a method that automatically extracts inter-linked domain-specific taxonomies from the DBpedia ontology. These taxonomies are then used to establish semantic paths between items from different domains, forming the abovementioned instance networks.
- For **measuring semantic relatedness between items from different domains** (RQ3), we have evaluated a number of text- and graph-based ranking algorithms, and have shown that a proposed simple weight spreading algorithm highly correlated with human judgments of cross-domain item semantic relatedness.
- With respect to **enhancing cross-domain recommendations based on user preferences by considering items semantically related** (RQ4), we have investigated two effective approaches, which combine content-based recommendations with semantic similarities, and which provide personalized graph-based semantic similarities. Empirically evaluating these approaches, we have confirmed that exploiting semantic relatedness was crucial for finding the relevant items for a user in a context-aware cross-domain recommendation setting.

We instantiated our framework for the particular application of suggesting musicians related to POIs, and conducted several experiments to evaluate the different stages of the approach. On a dataset of 25 POIs from major European cities, and through 3 user studies, we proved that musicians automatically extracted from DBpedia and incorporated into instance-level networks, which were top ranked by the proposed weight spreading algorithm, were judged by users as relevant for the input POIs in 70% to 90% of the evaluation cases. In this context, we showed that in general the users' assessments were not influenced by personal music preferences. We also proved that semantically enhanced algorithms outperform purely content-based algorithms in the

task of providing personalized suggestions of musicians for a given POI, thus confirming that users give more importance to semantic relatedness over personal preferences.

5.2 Future work

The work done in this thesis represents initial steps towards a generic and flexible semantic-based framework to support cross-domain item recommendations. There are many challenging research issues that have to be addressed. In the following we discuss those we consider as most important:

- **Addressing alternative domains.** The implementation and evaluation of the framework were performed in the particular application of suggesting musicians related to places of interest, which is a relevant context-aware recommendation situation.
  In the future we plan to instantiate our framework in other domains. Specifically, we have started to explore entertainment domains –such as movies, music, and books– (Cantador et al., 2013; Fernández-Tobías et al., 2013) in which user preferences are easier to be obtained, e.g. from user profiles available in online social networks.
- **Considering arbitrary semantic relations.** As discussed in Section 3.1.1, so far our framework does not use arbitrary semantic relations that may exist between items in different domains, e.g. "Gustav Mahler *was the director of* the Vienna State Opera". Since relations of this type are not captured by explicit properties in DBpedia, we plan to exploit other structured knowledge bases. In particular, we have started to investigate on the use of tools like RelFinder (Heim et al., 2009) to search for non-trivial relations in Linked Data repositories. For the abovementioned entertainment domains, examples of semantic relations we have automatically extracted are "The American R&B, soul, gospel and funk singer Philip Bailey *appears in* the Full Metal Jacket movie", "The book The Short-timers *was the basis* of the movie Full Metal Jacket", and "The movie Platoon *showed the horrors* of Vietnam war, which *is the ultimate farce in* the movie Full Metal Jacket."
  Due to the difficulty of finding such relations in a generic way for an arbitrary cross-domain application, we also plan to investigate how to exploit semantic annotation tools such as DBpedia Spotlight (Mendes et al., 2011) and ReVerb (Etzioni et al., 2011) to automatically identify semantic concepts and relations in text corpora, such as Wikipedia and the ClueWeb dataset[11]. Examples of POI-musician semantic relations we have already extracted from the above text collections are "Gustav Mahler *became director of* the Vienna State Opera", "Ketama (a Flamenco fusion band) *was formed in* Madrid, and *played a concert at* Las Ventas", and Brian May (an English Rock musician) *performed "God Save the Queen" from the roof of* Buckingham Palace. We shall conduct evaluations to assess the quality and value of these relations.
- **Automatic initialization of relevance values.** In the performed implementation of our framework, we manually set all relevance values of the classes and class-level relations, and in the personalized graph-based ranking algorithms (Section 3.4.2), we set the initial relevance values of some classes according to the users' preferences.
  In the future, we shall investigate other automatic methods to initialize the above relevance values. For example, the relevance values of the classes may be set by a concept frequency-based heuristic that computes TF-IDF weights of concepts within the DBpedia ontology. The relevance of relations, on the other hand, may be set by a strategy that penalizes relations in long semantic paths between instances, which were found as less meaningful by the users in the conducted experiments.

[11] ClueWeb'12 dataset, http://lemurproject.org/clueweb12/

- **Exploiting cross-domain user preferences.** The recommendation methods presented in Section 3.4 integrate semantic relatedness scores with user preferences in the target domain. In the conducted experiments, for the POI-to-musician recommendation case study, the users stated their favorite music genres, but their preferences about POIs, and the Architecture, Art, and History domains were ignored. We plan to investigate recommendation methods that also take into account user preferences on source domains. For example, we could set prior relevance values to classes and instances in the semantic networks, as we already did in the target domain. Another option is to integrate item-to-item semantic relatedness values computed as described in Section 3.3 into cross-domain matrix factorization algorithms, such as TagCDCF (Shi et al., 2011).

References

Adomavicius, G., Sankaranarayanan, R., Sen, S., & Tuzhilin, A. (2005). Incorporating Contextual Information in Recommender Systems Using a Multidimensional Approach. *ACM Transactions on Information Systems*, *23*(1), 103–145.

Adomavicius, G., & Tuzhilin, A. (2005). Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering*, *17*(6), 734–749.

Adomavicius, G., & Tuzhilin, A. (2011). Context-Aware Recommender Systems. In F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), *Recommender Systems Handbook*, pp. 217–253. Springer.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. G. (2007). DBpedia: A Nucleus for a Web of Open Data. In *Proceedings of the 6th International Semantic Web Conference (ISWC 2007)*, pp. 722–735.

Baeza-Yates, R. A., & Ribeiro-Neto, B. A. (1999). *Modern Information Retrieval*. Addison-Wesley.

Baltrunas, L., & Ricci, F. (2009). Context-based Splitting of Item Ratings in Collaborative Filtering. In *Proceedings of the 3rd ACM Conference on Recommender Systems (RecSys 2009),* pp. 245–248.

Baltrunas, L., & Ricci, F. (2013). Experimental Evaluation of Context-dependent Collaborative Filtering Using Item Splitting. *User Modeling and User-Adapted Interaction*, February 2013, 1–28.

Berkovsky, S., Kuflik, T., & Ricci, F. (2007). Cross-Domain Mediation in Collaborative Filtering. In *Proceedings of the 11th international conference on User Modeling (UM 2007)*, pp. 355–359.

Bizer, C., Heath, T., & Berners-Lee, T. (2009). Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, *5*(3), 1–22.

Brin, S., & Page, L. (1998). The Anatomy of a Large-scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems*, *30*(1-7), 107–117.

Budanitsky, A., & Hirst, G. (2001). Semantic Distance in WordNet: An Experimental, Application-oriented Evaluation of Five Measures. In *Porceedings of the Workshop on WordNet and other lexical resources, at the 2nd meeting of the North American Chapter of the Association for Computational Linguistics*.

Burke, R. D. (2000). Knowledge-Based Recommender Systems. *Encyclopedia of Library and Information Systems*. Marcel Dekker.

Burke, R. D. (2002). Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, *12*(4), 331–370.

Cantador, I., Bellogín, A., & Vallet, D. (2010). Content-based Recommendation in Social Tagging Systems. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys 2010),* pp. 237–240.

Cantador, I., Castells, P., & Bellogín, A. (2011). An Enhanced Semantic Layer for Hybrid Recommender Systems: Application to News Recommendation. *International Journal on Semantic Web and Information Systems*, *7*(1), 44–78.

Cantador, I., Fernández-Tobías, I., Bellogín, A, Kosinski, M., Stillwell, D. (2013). Relating Personality Types with User Preferences in Multiple Entertainment Domains. In *Proceedings of the 1st Workshop on Emotions and Personality in Personalized Services (EMPIRE 2013), at the 21st Conference on User Modeling, Adaptation and Personalization (UMAP 2013)*, pp.

Cremonesi, P., Tripodi, A., & Turrin, R. (2011). Cross-Domain Recommender Systems. In *IEEE 11th International Conference on Data Mining Workshops (ICDMW 2011)*, pp. 496–503.

Crestani, F. (1997). Application of Spreading Activation Techniques in Information Retrieval. *Artificial Intelligence Review*, *11*(6), 453–482.

Croft, W. B. (2002). Combining Approaches to Information Retrieval. In W. B. Croft (Ed.), *Advances in Information Retrieval*, vol. 7, pp. 1–36. Springer.

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., & Harshman, R. A. (1990). Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, *41*(6), 391–407.

Desrosiers, C., & Karypis, G. (2011). A Comprehensive Survey of Neighborhood-based Recommendation Methods. In F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), *Recommender Systems Handbook*, pp. 107–144. Springer.

Dey, A. K. (2001). Understanding and Using Context. *Personal and Ubiquitous Computing*, *5*(1), 4–7.

Etzioni, O., Fader, A., Christensen, J., Soderland, S., & Mausam. (2011). Open Information Extraction: The Second Generation. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011),* pp. 3–10.

Felfernig, A., Friedrich, G., Jannach, D., & Zanker, M. (2011). Developing Constraint-based Recommenders. In F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), *Recommender Systems Handbook*, pp. 187–215. Springer.

Felfernig, A., Isak, K., Szabo, K., & Zachar, P. (2007). The VITA Financial Services Sales Support Environment. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI 2007)*, pp. 1692–1699.

Felfernig, A., & Shchekotykhin, K. M. (2006). Debugging User Interface Descriptions of Knowledge-based Recommender Applications. In *Proceedings of the 11th International Conference on Intelligent User Interfaces (IUI 2006)*, pp. 234–241.

Fernández-Tobías, I., Cantador, I., Plaza, L. (2013) An Emotion Dimensional Model Based on Social Tags: Crossing Folksonomies and Enhancing Recommendations. In *Proceedings of the 14th International Conference on Electronic Commerce and Web Technologies (EC-Web 2013)*, pp. 88-100.

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., & Ruppin, E. (2002). Placing Search in Context: The Concept Revisited. *ACM Transactions on Information Systems*, *20*(1), 116–131.

Gabrilovich, E., & Markovitch, S. (2007). Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007),* pp. 1606–1611.

Heim, P., Hellmann, S., Lehmann, J., Lohmann, S., & Stegemann, T. (2009). RelFinder: Revealing Relationships in RDF Knowledge Bases. In *Proceedings of the 4th International Conference on Semantic and Digital Media Technologies (SAMT 2009),* pp. 182–187.

Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl, J. (1999). An Algorithmic Framework for Performing Collaborative Filtering. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1999),* pp. 230–237.

Hotho, A., Jäschke, R., Schmitz, C., & Stumme, G. (2006). Information Retrieval in Folksonomies: Search and Ranking. In *Proceedings of the 3rd European Semantic Web Conference (ESWC 2006),* pp. 411–426.

Jannach, D., Zanker, M., & Fuchs, M. (2009). Constraint-Based Recommendation in Tourism: A Multiperspective Case Study. *Journal of Information Technology & Tourism*, *11*(2), 139–155.

Jiang, J. J., & Conrath, D. W. (1997). Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. *Co*RR, *cmp-lg/970*.

Kaminskas, M., & Ricci, F. (2011). Location-Adapted Music Recommendation Using Tags. In *Proceedings of the 19th International Conference on User Modeling, Adaption and Personalization (UMAP 2011),* pp. 183–194.

Karatzoglou, A., Amatriain, X., Baltrunas, L., & Oliver, N. (2010). Multiverse Recommendation: N-dimensional Tensor Factorization for Context-aware Collaborative Filtering. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys 2010),* pp. 79–86.

Kleinberg, J. M. (1999). Hubs, Authorities, and Communities. *ACM Computing Surveys*, *31*(4es), 5.

Koren, Y., & Bell, R. M. (2011). Advances in Collaborative Filtering. In F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), *Recommender Systems Handbook*, pp. 145–186. Springer.

Leacock, C., & Chodorow, M. (1998). Combining Local Context and WordNet Similarity for Word Sense Identification. *MIT Press*, pp. 265–283.

Lee, M. D., Pincombe, B., & Welsh, M. (2005). An Empirical Evaluation of Models of Text Document Similarity. In *Proceedings of the 27th Annual Cognitive Science Conference (CogSci 2005*), pp. 1254–1259.

Li, B. (2011). Cross-Domain Collaborative Filtering: A Brief Survey. In *IEEE 23rd International Conference on Tools with Artificial Intelligence (ICTAI 2011),* pp. 1085–1086.

Lin, D. (1998). An Information-Theoretic Definition of Similarity. In *Proceedings of the 15th International Conference on Machine Learning (ICML 1998),* pp. 296–304.

Loizou, A. (2009). How to Recommend Music to Film Buffs: Enabling the Provision of Recommendations from Multiple Domains. *Master thesis, University of Southampton*.

Lops, P., de Gemmis, M., & Semeraro, G. (2011). Content-based Recommender Systems: State of the Art and Trends. In F. Ricci, L. Rokach, B. Shapira, & P. B. Kantor (Eds.), *Recommender Systems Handbook*, pp. 73–105. Springer.

Mahmood, T., & Ricci, F. (2007). Learning and Adaptivity in Interactive Recommender Systems. In *Proceedings of the 9th International Conference on Electronic Commerce (ICEC 2007),* pp. 75–84.

Mendes, P. N., Jakob, M., García-Silva, A., & Bizer, C. (2011). DBpedia Spotlight: Shedding Light on the Web of Documents. In *Proceedings the 7th International Conference on Semantic Systems (I-SEMANTICS 2011),* pp. 1–8.

Middleton, S. E., Roure, D. De, & Shadbolt, N. R. (2004). Ontology-based Recommender Systems. In S. Staab & R. Studer (Eds.), *Handbook on Ontologies*, pp. 498–577. Springer.

Miller, G. A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM (CACM 1995)*, *38*(11), 39–41.

Miller, G. A., & Charles, W. G. (1991). Contextual Correlates of Semantic Similarity. *Language & Cognitive Processes*, *6*(1), 1–28.

Milne, D., & Witten, I. H. (2008). An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links. In *Proceedings of the 1st AAAI Workshop on Wikipedia and Artificial Intelligence*.

Mobasher, B., Jin, X., & Zhou, Y. (2003). Semantically Enhanced Collaborative Filtering on the Web. In *Proceedings of the 1st European Web Mining Forum (EMWF 2003),* pp. 57–76.

Oku, K., Nakajima, S., Miyazaki, J., & Uemura, S. (2006). Context-Aware SVM for Context-Dependent Information Recommendation. In *Proceedings of the 7th International Conference on Mobile Data Management (MDM 2006),* pp. 109-112.

Panniello, U., Tuzhilin, A., Gorgoglione, M., Palmisano, C., & Pedone, A. (2009). Experimental Comparison of Pre- vs. Post-filtering Approaches in Context-aware Recommender Systems. In *Proceedings of the 3rd ACM Conference on Recommender Systems (RecSys 2009),* pp. 265–268.

Passant, A. (2010). Measuring Semantic Distance on Linking Data and Using it for Resources Recommendations. In *Proceedings of the AAAI 2010 Spring Symposium: Linked Data Meets Artificial Intelligence*.

Rada, R., Mili, H., Bicknell, E., & Blettner, M. (1989). Development and Application of a Metric on Semantic Nets. *IEEE Transactions on Systems, Man, and Cybernetics*, *19*(1), 17–30.

Rendle, S., Gantner, Z., Freudenthaler, C., & Schmidt-Thieme, L. (2011). Fast Context-aware Recommendations with Factorization Machines. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2011),* pp. 635–644.

Resnik, P. (1995). Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI 1995),* pp. 448–453.

Richardson, R., & Smeaton, A. F. (1995). Using WordNet in a Knowledge-based Approach to Information Retrieval.

Rubenstein, H., & Goodenough, J. B. (1965). Contextual Correlates of Synonymy. *Communications of the ACM*, *8*(10), 627–633.

Seco, N., Veale, T., & Hayes, J. (2004). An Intrinsic Information Content Metric for Semantic Similarity in WordNet. In *Proceedings of the 16th Eureopean Conference on Artificial Intelligence (ECAI 2004),* pp. 1089–1090.

Shi, Y., Larson, M., & Hanjalic, A. (2011). Tags as Bridges between Domains: Improving Recommendation with Tag-Induced Cross-Domain Collaborative Filtering. In *Proceedings of the 19th International Conference on User Modeling, Adaption and Personalization (UMAP 2011),* pp. 305–316.

Strube, M., & Ponzetto, S. P. (2006). WikiRelate! Computing Semantic Relatedness Using Wikipedia. In *Proceedings of the 21st National Conference on Artificial Intelligence and the 18th Innovative Applications of Artificial Intelligence Conference,* pp. 1419–1424.

Stupar, A., & Michel, S. (2011). Picasso - To Sing, You Must Close Your Eyes and Draw. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2011),* pp. 715–724.

Tiroshi, A., & Kuflik, T. (2012). Domain Ranking for Cross Domain Collaborative Filtering. In *Proceedings of the 20th International Conference on User Modeling, Adaption and Personalization (UMAP 2012),* pp. 328–333.

Vander Wal, T. (2007). Folksonomy Coinage and Definition. September 16, 2013. On-line available at http://vanderwal.net/folksonomy.html

White, S., & Smyth, P. (2003). Algorithms for Estimating Relative Importance in Networks. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2003),* pp. 266–275.

Winoto, P., & Tang, T. (2008). If You Like the Devil Wears Prada the Book, Will You also Enjoy the Devil Wears Prada the Movie? A Study of Cross-Domain Recommendations. *New Generation Computing*, *26*, 209–225.

Wu, Z., & Palmer, M. S. (1994). Verb Semantics and Lexical Selection. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics (ACL 1994),* pp. 133–138.

Appendix A. Rank correlations between semantic relatedness algorithms

The following table shows the rank correlations –computed with the Spearman's coefficient– between the semantic relatedness metrics used in the experiment presented in Section 4.1.1.

	HITS	HITS (undirected)	PageRank	PageRank (undirected)	Spreading	Ratio	Common features	Distinctive features	Cosine	Cosine LSA (K = 10)	Cosine LSA (K = 100)	Dice	Dice LSA (K = 10)	Dice LSA (K = 100)	Jaccard	Jaccard LSA (K = 10)	Jaccard LSA (K = 100)	Overlap	Overlap LSA (K = 10)	Overlap LSA (K = 100)
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	1.00																			
2	0.66	1.00																		
3	0.89	0.82	1.00																	
4	0.84	0.77	0.86	1.00																
5	0.15	0.10	0.11	0.35	1.00															
6	0.08	0.07	0.11	0.09	0.05	1.00														
7	0.17	0.18	0.28	0.19	0.08	0.77	1.00													
8	-0.19	-0.20	-0.34	-0.20	-0.07	-0.24	-0.78	1.00												
9	0.28	0.34	0.36	0.31	0.30	0.45	0.52	-0.37	1.00											
10	0.27	0.44	0.40	0.33	0.19	0.29	0.42	-0.36	0.54	1.00										
11	0.28	0.39	0.36	0.31	0.29	0.44	0.47	-0.29	0.69	0.73	1.00									
12	0.26	0.31	0.33	0.28	0.30	0.46	0.52	-0.36	0.99	0.54	0.69	1.00								
13	0.17	0.33	0.30	0.22	0.17	0.31	0.44	-0.35	0.53	0.91	0.68	0.54	1.00							
14	0.28	0.40	0.36	0.29	0.26	0.42	0.44	-0.26	0.64	0.72	0.97	0.65	0.72	1.00						
15	0.26	0.31	0.33	0.28	0.30	0.46	0.52	-0.36	0.99	0.54	0.69	1.00	0.54	0.65	1.00					
16	0.17	0.33	0.30	0.22	0.17	0.31	0.44	-0.35	0.53	0.91	0.68	0.54	1.00	0.72	0.54	1.00				
17	0.28	0.40	0.36	0.29	0.26	0.42	0.44	-0.26	0.64	0.72	0.97	0.65	0.72	1.00	0.65	0.72	1.00			
18	0.34	0.40	0.42	0.36	0.30	0.40	0.50	-0.38	0.96	0.53	0.65	0.91	0.48	0.59	0.91	0.48	0.59	1.00		
19	0.36	0.46	0.46	0.40	0.12	0.12	0.26	-0.27	0.36	0.72	0.49	0.32	0.42	0.41	0.32	0.42	0.41	0.42	1.00	
20	0.21	0.27	0.27	0.26	0.28	0.39	0.44	-0.30	0.62	0.56	0.83	0.60	0.43	0.67	0.60	0.43	0.67	0.62	0.55	1.00

Appendix B. Semantic relatedness performance for different types of POIs

The following table shows the complete performance results of the selected semantic relatedness algorithms for different types of POIs.

	MAP	P@1	P@2	P@3	P@4	P@5	nDCG
Castles and palaces (6)							
Random	0.013	0.000	0.028	0.037	0.028	0.033	0.239
Cosine	0.271	0.333	0.500	0.500	0.542	0.533	0.561
Cosine LSA (k = 10)	0.174	0.333	0.583	0.500	0.500	0.467	0.495
Cosine LSA (k = 100)	0.220	0.500	0.417	0.444	0.500	0.433	0.541
Overlap	0.301	0.500	0.500	0.556	0.542	0.533	0.577
Overlap LSA (k = 10)	0.141	0.333	0.417	0.444	0.375	0.367	0.436
Overlap LSA (k = 100)	0.273	**1.000**	0.667	0.611	0.542	0.500	0.575
HITS	0.101	0.167	0.167	0.222	0.167	0.167	0.372
HITS (undirected)	0.109	0.000	0.083	0.167	0.292	0.233	0.370
PageRank	0.101	0.167	0.167	0.222	0.167	0.167	0.372
PageRank (undirected)	0.217	0.667	0.417	0.389	0.417	0.367	0.503
Spreading	**0.363**	0.833	**0.750**	**0.778**	**0.750**	**0.800**	**0.593**
Music venues (4)							
Random	0.014	0.000	0.000	0.028	0.021	0.017	0.284
Cosine	0.376	0.750	0.875	0.833	0.875	0.850	0.752
Cosine LSA (k = 10)	0.392	**1.000**	**1.000**	**1.000**	0.875	0.850	0.716
Cosine LSA (k = 100)	0.414	0.750	0.875	0.917	**0.938**	**0.900**	0.749
Overlap	0.395	0.750	0.875	0.750	0.750	0.800	0.745
Overlap LSA (k = 10)	0.120	0.500	0.375	0.333	0.250	0.300	0.436
Overlap LSA (k = 100)	0.249	0.500	0.625	0.583	0.625	0.550	0.581
HITS	0.358	0.750	0.625	0.583	0.500	0.500	0.647
HITS (undirected)	0.379	0.750	0.750	0.833	0.875	0.850	0.667
PageRank	0.358	0.750	0.625	0.583	0.500	0.500	0.647
PageRank (undirected)	0.357	0.750	0.625	0.667	0.625	0.600	0.686
Spreading	**0.464**	0.750	0.750	0.750	0.750	0.750	**0.766**
Religious places (8)							
Random	0.009	0.000	0.000	0.000	0.021	0.017	0.220
Cosine	0.314	0.500	0.625	0.625	0.625	0.550	0.645
Cosine LSA (k = 10)	0.295	0.625	0.563	0.625	0.656	0.625	0.565
Cosine LSA (k = 100)	0.127	0.125	0.188	0.125	0.156	0.175	0.421
Overlap	0.250	0.625	0.438	0.375	0.438	0.475	0.592
Overlap LSA (k = 10)	0.054	0.250	0.188	0.167	0.125	0.125	0.298
Overlap LSA (k = 100)	0.029	0.000	0.063	0.042	0.031	0.025	0.275
HITS	0.241	0.375	0.500	0.375	0.344	0.300	0.489
HITS (undirected)	0.193	0.125	0.063	0.167	0.219	0.200	0.420
PageRank	0.241	0.375	0.500	0.375	0.344	0.300	0.489
PageRank (undirected)	0.360	0.750	0.563	0.542	0.500	0.475	0.572
Spreading	**0.457**	**1.000**	**1.000**	**0.833**	**0.844**	**0.725**	**0.661**

	MAP	P@1	P@2	P@3	P@4	P@5	nDCG
Other POIs (7)							
Random	0.028	0.095	0.071	0.111	0.119	0.114	0.286
Cosine	0.319	0.714	0.714	0.667	0.714	**0.714**	0.616
Cosine LSA (k = 10)	0.215	0.857	0.571	0.571	0.607	0.543	0.501
Cosine LSA (k = 100)	0.297	0.714	0.714	0.619	0.643	0.629	0.559
Overlap	0.296	0.714	0.714	0.619	0.607	0.600	0.604
Overlap LSA (k = 10)	0.167	0.714	0.571	0.524	0.536	0.486	0.481
Overlap LSA (k = 100)	0.188	0.429	0.429	0.476	0.464	0.514	0.485
HITS	0.126	0.429	0.286	0.286	0.357	0.286	0.452
HITS (undirected)	0.118	0.286	0.286	0.286	0.286	0.229	0.444
PageRank	0.126	0.429	0.286	0.286	0.357	0.286	0.452
PageRank (undirected)	0.215	0.571	0.571	0.429	0.429	0.400	0.563
Spreading	**0.379**	**1.000**	**0.857**	**0.810**	**0.786**	0.686	**0.681**